How initialization helps the training process

Zhihan Hu

February 2025

1 Introduction

In this document, I am going to introduce the theoretical basis for Xavier and He initialization. And extends the idea of initialization schemes to a general methodology.

2 Difficulties when training DL models

Around 10 years ago, when deep learning models first gained popularity due to their ability to represent complex and non-linear relationships, a significant challenge emerged alongside their rise: **the problem of vanishing gradients.**.

This issue is primarily due to **Activation function issues**. The randomly initialized weights may easily lead the activation values fall onto the saturation regimen of the activation function curve, where the gradient is close to 0. And thus causing the training process to slow down or even stall.

In the paper *Understanding the difficulty of training deep feedforward neural networks*, Xavier et al has shown that by simply initializing the weights using $U\left(-\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}\right)$, where n is the number of neurons on the previous layer, will cause a model which uses sigmoid as activation function to saturate immediately. For models using tanh and softsign as activation function, the training process will be faster, however, there is still instability on the loss due to saturation of the activation functions.

3 Methodologies

Before talking about the solution, we need an assumption here:

- Suppose the input dataset is already normalized such that the initial average is 0 and the variance is scaled down.

With such an assumption, we can assume that the average of the output of a hidden layer will be 0. When the output of each neuron is 0, it will fall onto the regimen where the gradient is unity (1), which is good for the training process. However, the reason this normalization is not enough is that the variance of each output neuron unit, after applied by the weight matrix, is not constrained. For example, if the variance is 5, then even though the neuron's average value is 0, there is a large potential that the actual value becomes larger than 5, leading to a saturate of the activation value. And this intuitively lead to the first methodology: **We want the variance of the activation layer to be stable and equal to the variance of the input training data**.

Let me explain it a little bit further to avoid confusions. The variance of input training data, $Var[X]$, is defined as the expected variance across all features. Since there is no prior information on the data, we need to assume that the variance of each feature is roughly equal, i.e, $Var[X_1] = Var[X_2] = ... = Var[X_n] = Var[X]$. Similarly, for each activation value on an activation layer, we can assume that the variance of each value is the same, i.e, $Var[A_1] = Var[A_2] = ... = Var[A_m] = Var[A]$. We want this relation to hold layer-wisely, and thus we can have

$$Var[A^{l-1}] = Var[Z^l] \tag{1}$$

The first methodology is about forward propagation, we also need a constrain on the variance of the gradient backwardly. i.e

$$Var[\delta^{l-1}] = Var[\delta^l] \tag{2}$$

Such that the gradient will not suddenly switch to a high or low value, which leads to saturation.

4 Xavier initialization

Now let's begin to derive how Xavier initialization work.

The i^{th} neuron of layer l is computed as

$$Z_i^{(l)} = \sum_{k=1}^{n_{l-1}} W_{ik}^{(l)} A_k^{(l-1)} \tag{3}$$

Where n_{l-1} is the number of neurons on layer l-1. Then we have

$$Var[Z_i^{(l)}] = \sum_{k=1}^{n_{l-1}} Var[W_{ik}^{(l)} A_k^{(l-1)}] \tag{4}$$

Since $W^{(l)}$ and $A^{(l-1)}$ are independent, as $A^{(l-1)}$ only depends on $W^{(l-1)}$, we can have

$$Var[Z_i^{(l)}] = \sum_{k=1}^{n_{l-1}} Var[W_{ik}^{(l)}] Var[A_k^{(l-1)}] \tag{5}$$

The equation 5 can be rewritten into

$$Var[Z_i^{(l)}] = n_{(l-1)} Var[W^{(l)}] Var[A^{(l-1)}] \tag{6}$$

By equation 1, to make equation 6 hold, we have

$$n_{(l-1)} Var[W^{(l)}] = 1 \tag{7}$$

And thus

$$Var[W^{(l)}] = \frac{1}{n_{l-1}} \tag{8}$$

In the backward propagation, we have

$$\delta^{l-1} = (W^{(l-1)})^T \delta^{(l)} \circ \sigma'(Z^{(l-1)}) \tag{9}$$

Where $\delta^{l-1} = \frac{\delta L}{\delta Z^{(l-1)}}$

Since we have normalized our data, we can assume that $\sigma'(Z^{(l-1)})$ locates at near 0, whose gradient is unity, and thus

$$\delta^{l-1} = (W^{(l-1)})^T \delta^{(l)} \tag{10}$$

Since

$$\delta_{ij}^{l-1} = \sum_{k=1}^{n_l} W_{ik}^{(l)T} \delta_{kj}^{(l)} \tag{11}$$

Then, we can have

$$Var[\delta_{ij}^{l-1}] = n_l Var[W_{ik}^{(l)T}] Var[\delta_{kj}^{(l-1)}] = n_l Var[W^{(l)}] Var[\delta^l] \tag{12}$$

Based on equation 2, to make equation 12 holds, we get

$$n_l Var[W^{(l)}] = 1 \tag{13}$$

And thus

$$Var[W^{(l)}] = \frac{1}{n_l} \tag{14}$$

Now, we can average the variance of $W^{(l)}$ forwardly and backwardly to get an integrated variance:

$$Var[W^{(l)}] = \frac{2}{n_{l-1} + n_l} \tag{15}$$

Since we obtain the variance, we can now find the suitable distribution for initializing $W^{(l)}$:

Using uniform

$$W^{(l)} = U[-\frac{\sqrt{6}}{\sqrt{n_{l-1} + n_l}}, \frac{\sqrt{6}}{\sqrt{n_{l-1} + n_l}}] \tag{16}$$

Normal

$$W^{(l)} = N[0, \frac{2}{n_{l-1} + n_l}] \tag{17}$$

5 He initialization

One of the assumption made in Xavier initialization is that the activation function is symmetric around the origin. However, one of the most widely used activation functions, the ReLU, does not hold here. Additionally, when use ReLU, some of the activation value will become 0, and thus there should be some modification in the forward and backward requirement.

Forward propagation In the activation vector activated by ReLU, only around half of the activation values contribute to the variance, and thus the variance should decrease by 50% in order for the activated values variance stable, i.e

$$Var[A^{(l-1)}] = \frac{1}{2}Var[Z^{(l)}] \tag{18}$$

Then, we can have

$$\frac{1}{2}n_{l-1}Var[W^{(l)}] = 1 \tag{19}$$

And thus

$$Var[W^{(l)}] = \frac{2}{n_{l-1}} \tag{20}$$

For backward part, we can assume $E[\sigma'(Z^{(l-1)})] = \frac{1}{2}$, and then we have

$$Var[\delta^{(l-1)}] = n_l Var[W^{(l)}]Var[\delta^{(l)}] * \frac{1}{2} \tag{21}$$

And thus

$$Var[W^{(l)}] = \frac{2}{n_l} \tag{22}$$

However, based on He's research, when the activation is simply ReLU, the forward variance prioritize the training performance. And thus we have

$$Var[W^{(l)}] = \frac{2}{n_{l-1}} \tag{23}$$

Then, we have the distribution for initializing $W^{(l)}$

Uniform

$$W^{(l)} = U[-\frac{\sqrt{6}}{\sqrt{n_{l-1}}}, \frac{\sqrt{6}}{\sqrt{n_{l-1}}}] \tag{24}$$

Normal

$$W^{(l)} = N[0, \frac{2}{n_{l-1}}] \tag{25}$$

For ReLU with scaling, the backward variance will play a role here, we can average the backward and forward variance to obtain the integrated variance as

$$Var[W^{(l)}] = \frac{4}{n_{l-1} + n_l} \tag{26}$$

Then, we have the distribution for initializing $W^{(l)}$

Uniform

$$W^{(l)} = U[-\frac{\sqrt{12}}{\sqrt{n_{l-1}+n_l}}, \frac{\sqrt{12}}{\sqrt{n_{l-1}+n_l}}] \tag{27}$$

Normal

$$W^{(l)} = N[0, \frac{4}{n_{l-1}+n_l}] \tag{28}$$

6 Key Takeaway

If you're a practitioner, simply remember that

- If your activation function is **sigmoid, tanh or softsign**, use Xavier's initialization
  Uniform
  $$W^{(l)} = U[-\frac{\sqrt{6}}{\sqrt{n_{l-1}+n_l}}, \frac{\sqrt{6}}{\sqrt{n_{l-1}+n_l}}] \tag{29}$$
  Normal
  $$W^{(l)} = N[0, \frac{2}{n_{l-1}+n_l}] \tag{30}$$
- If your activation function is **ReLU**, Use He initialization
  Uniform
  $$W^{(l)} = U[-\frac{\sqrt{6}}{\sqrt{n_{l-1}}}, \frac{\sqrt{6}}{\sqrt{n_{l-1}}}] \tag{31}$$
  Normal
  $$W^{(l)} = N[0, \frac{2}{n_{l-1}}] \tag{32}$$
- If your activation function is **Scaled ReLU**, still use He initialization, but with some modification
  Uniform
  $$W^{(l)} = U[-\frac{\sqrt{12}}{\sqrt{n_{l-1}+n_l}}, \frac{\sqrt{12}}{\sqrt{n_{l-1}+n_l}}] \tag{33}$$
  Normal
  $$W^{(l)} = N[0, \frac{4}{n_{l-1}+n_l}] \tag{34}$$

If you're a geek, and is interested in understanding how these initialization schemes are derived, or even want create your own, or gain some technical barriers, please turn to the methodology parts for a general scheme for creating an initialization technique.