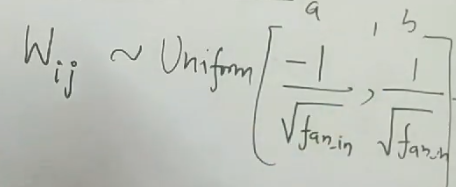
Weight initialization

Should not be small🡪 vanishing gradient problem

Should not be same(especially zero)🡪 don’t converge properly

Should have good variance🡪 good

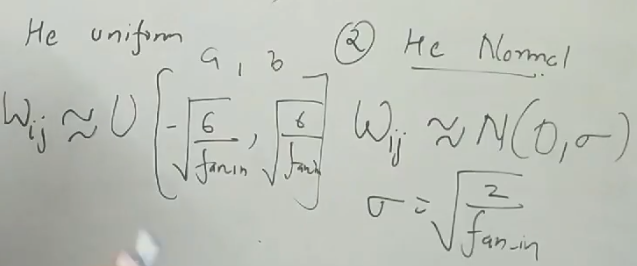
**Uniform initialization**: This method initializes the weights with values drawn from a uniform distribution. This can be useful for networks that use activation functions with a limited range, such as the sigmoid function.



**Xavier/Gorat initialization**: This method scales the weights by a factor that is inversely proportional to the square root of the number of inputs to the neuron. This helps to keep the variance of the activations constant across the layers.

|  |  |
| --- | --- |
| Xavier Normal | Xavior Uniform |
|  |  |

**He initialization**: This method scales the weights by a factor that is inversely proportional to the square root of the number of inputs to the neuron, but with a larger constant than in Xavier initialization. This is suitable for networks that use the ReLU activation function, as it helps to prevent the "dying ReLU" problem.



**LeCun initialization**: This method scales the weights by a factor that is inversely proportional to the square root of the number of inputs to the neuron, but with a smaller constant than in Xavier initialization. This is suitable for networks that use the hyperbolic tangent activation function.

**Orthogonal initialization:** This method initializes the weights as a random orthogonal matrix. This helps to prevent the vanishing or exploding gradients problem.