Nhat Le
CS 365
Lab D Neural Network

Testing Hyperparameters: Learning rate

Case: XOR gate, 2 inputs, 2 hidden nodes for 1 layer, 1 output

Data collection: Learning stops when the loss result is under 0.01 or the specified epoch is reached.

1. Learning rate = 0.3: Converged 6/10 times

Time	Epochs	Loss result
1	15,520	Converged <0.01
2	43,791	Converged <0.01
3	15,538	Converged <0.01
4	3m	Stuck at 0.477
5	3m	Stuck at 0.346
6	15,632	Converged <0.01
7	43,401	Converged <0.01
8	3m	Stuck at 0.477
9	3m	Stuck at 0.346
10	10,666	Converged <0.01

2. Learning rate of 0.6: Converged 5/10 times

Time	Epochs	Loss result
1	3m	Stuck at 0.346
2	3m	Stuck at 0.346
3	7,925	Converged <0.01
4	3m	Stuck at 0.346
5	22,036	Converged <0.01
6	21,940	Converged <0.01
7	5,206	Converged <0.01
8	3m	Stuck at 0.346
9	23,814	Converged <0.01
10	3m	Stuck at 0.477

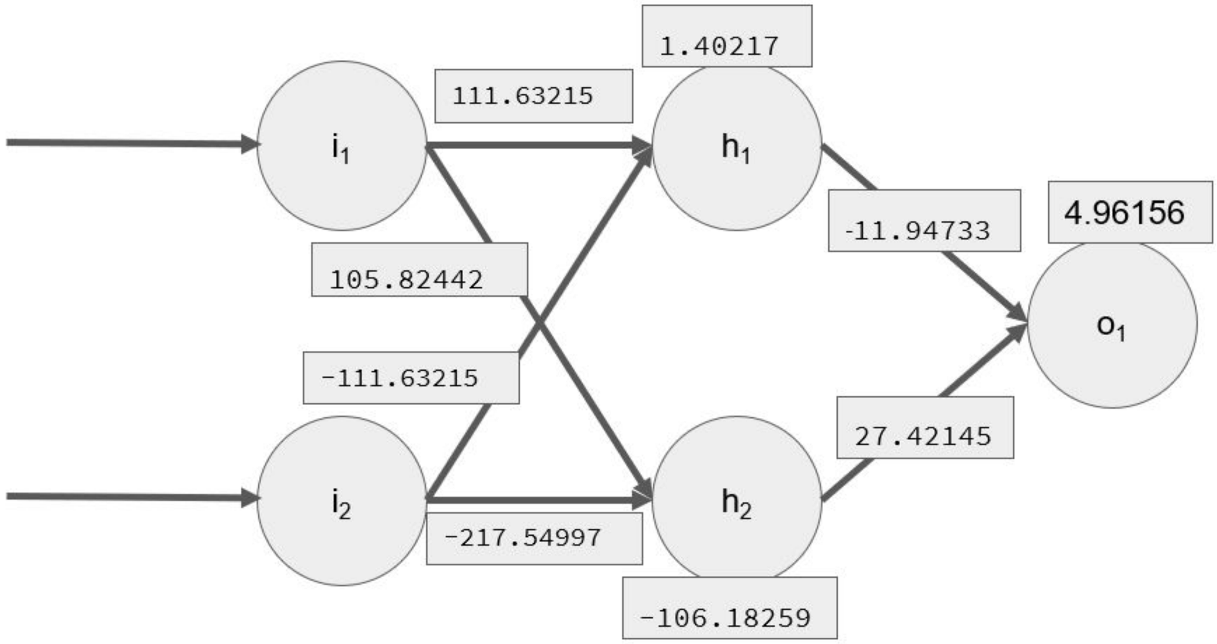
3. Learning rate = 0.1: Converged 5/10 times

Time	Epochs	Loss result
1	133,630	Converged <0.01
2	5m	Stuck at 0.346
3	5m	Stuck at 0.346
4	5m	Stuck at 0.477
5	48,885	Converged <0.01
6	133,257	Converged <0.01
7	132,795	Converged <0.01
8	49,488	Converged <0.01
9	5m	Stuck at 0.346
10	5m	Stuck at 0,477

4. Learning rate = 0.001:
   Converged 3/10 times

Time	Epochs	Loss result
1	3,194,238	Converged < 0.01
2	50m	Stuck at 0.34
3	50m	Stuck at 0.48
4	50m	Stuck at 0.48
5	50m	Stuck at 0.34
6	5,273,691	Converged < 0.01
7	50m	Stuck at 0.34
8	13,109,258	Converged
9	50m	Stuck at 0.34
10	50m	Stuck at 0.34

For all learning rates, the network does not always converge and it can get stuck at the loss of 0.34 and 0.48, which are potentially local minima in the loss graph making the weights and biases stuck at some points. When the network converges to loss of less than 0.01, higher learning rate generally needs fewer epochs than the lower ones. For example, when learning rate is 0.3, it takes tens of thousands of epochs to converge, but when the learning rate is 0.001, it takes millions of epochs to converge. This makes sense because the learning rate is like the size of your steps for Euler method in calculus with the gradient descent. Between the experimented learning rates, 0.3 is a good choice as it yields convergence for the highest number of times at a reasonable runtime. Neither higher learning rate like 0.6 nor especially lower like 0.001 has an overall advantage over 0.3.



This is a network converged after learning. It seems correct as it will yield a value very close to 1 when only one of the inputs is 1 and yield a value very close to 0 otherwise, matching with the behavior of XOR gate.