Adrien Protzel

Apples



def backward(self, grad): # grad = dL/dZ

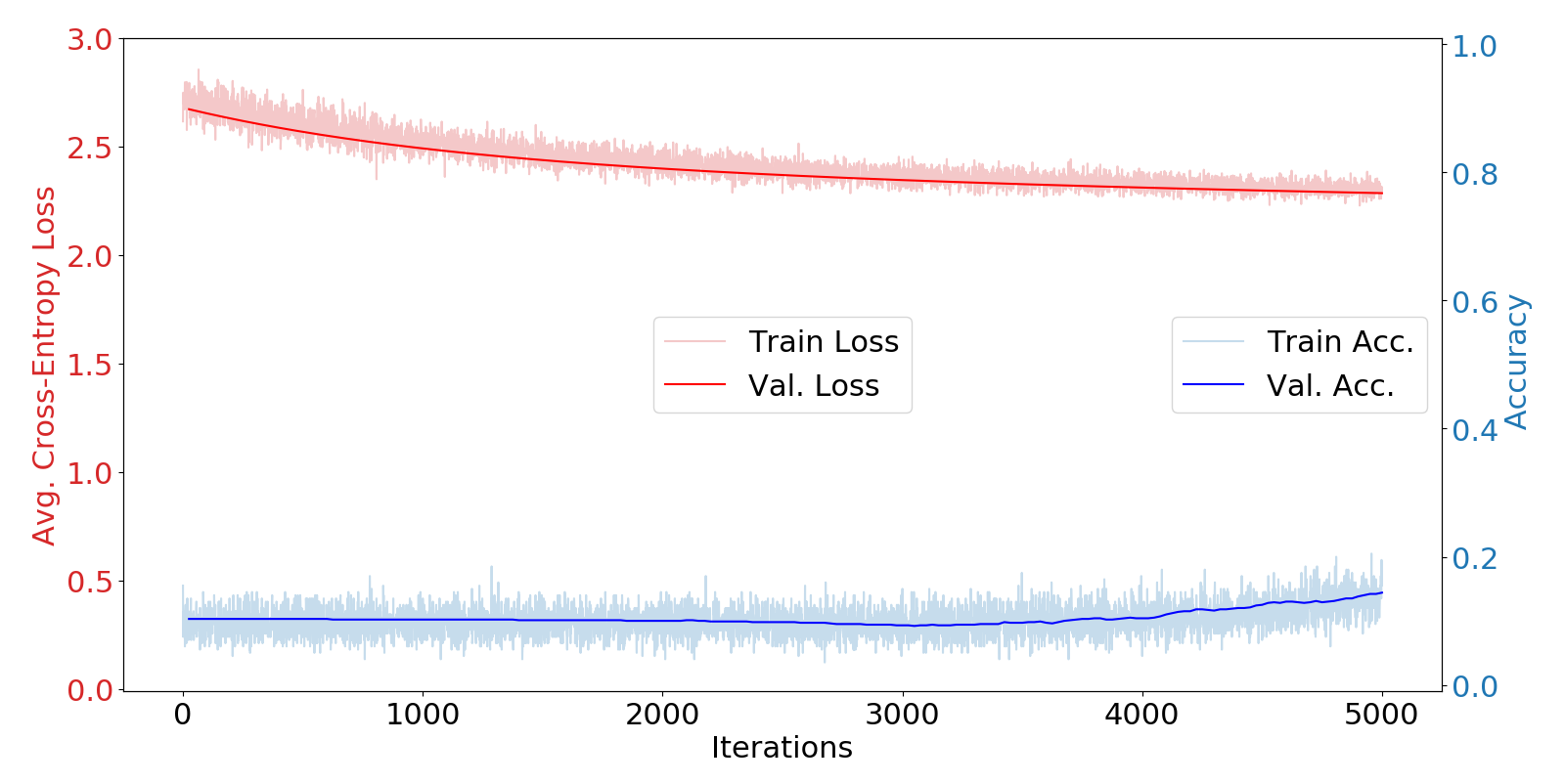
self.grad\_weights = self.input.T @ grad # = dL/dW

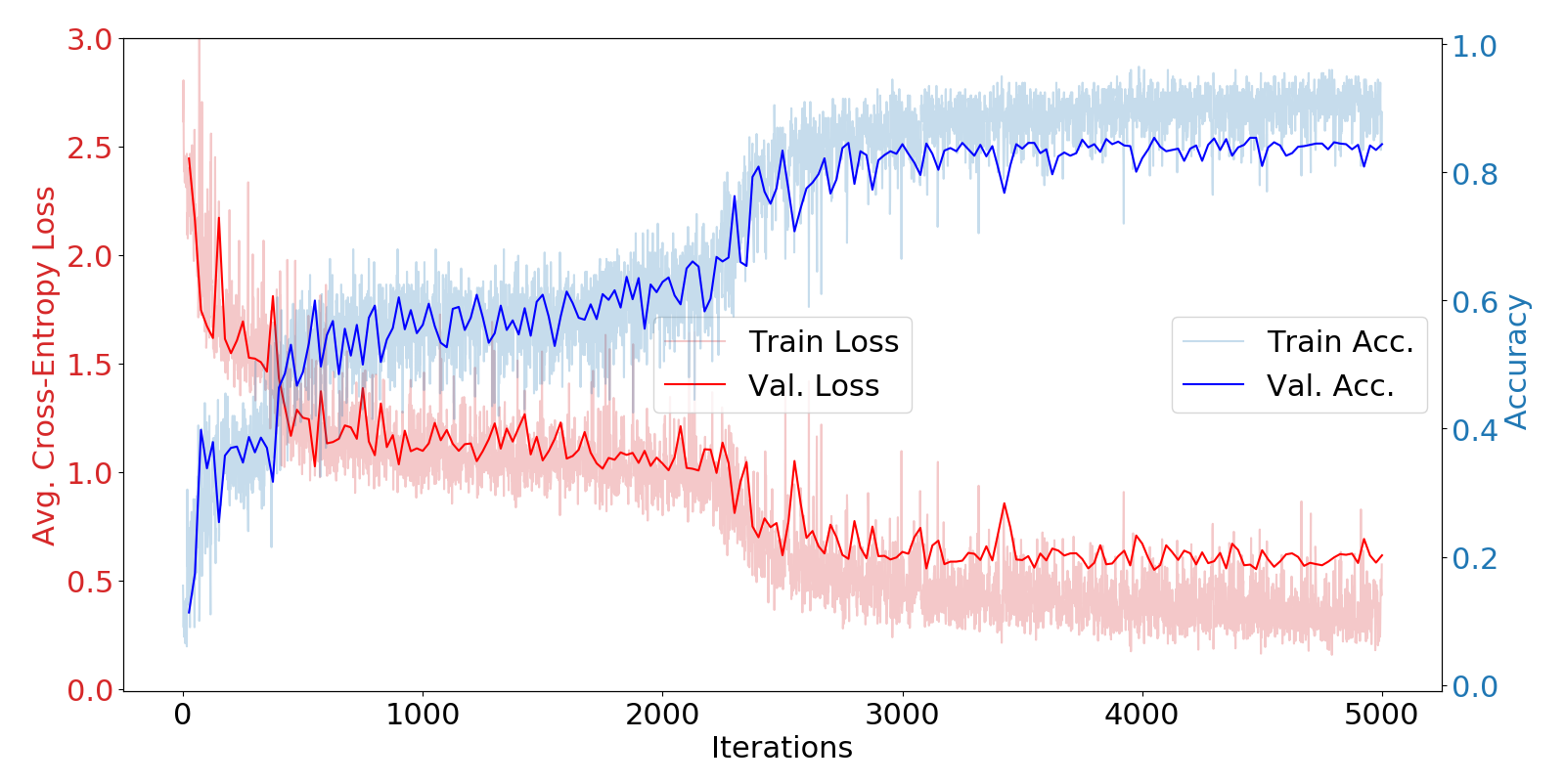
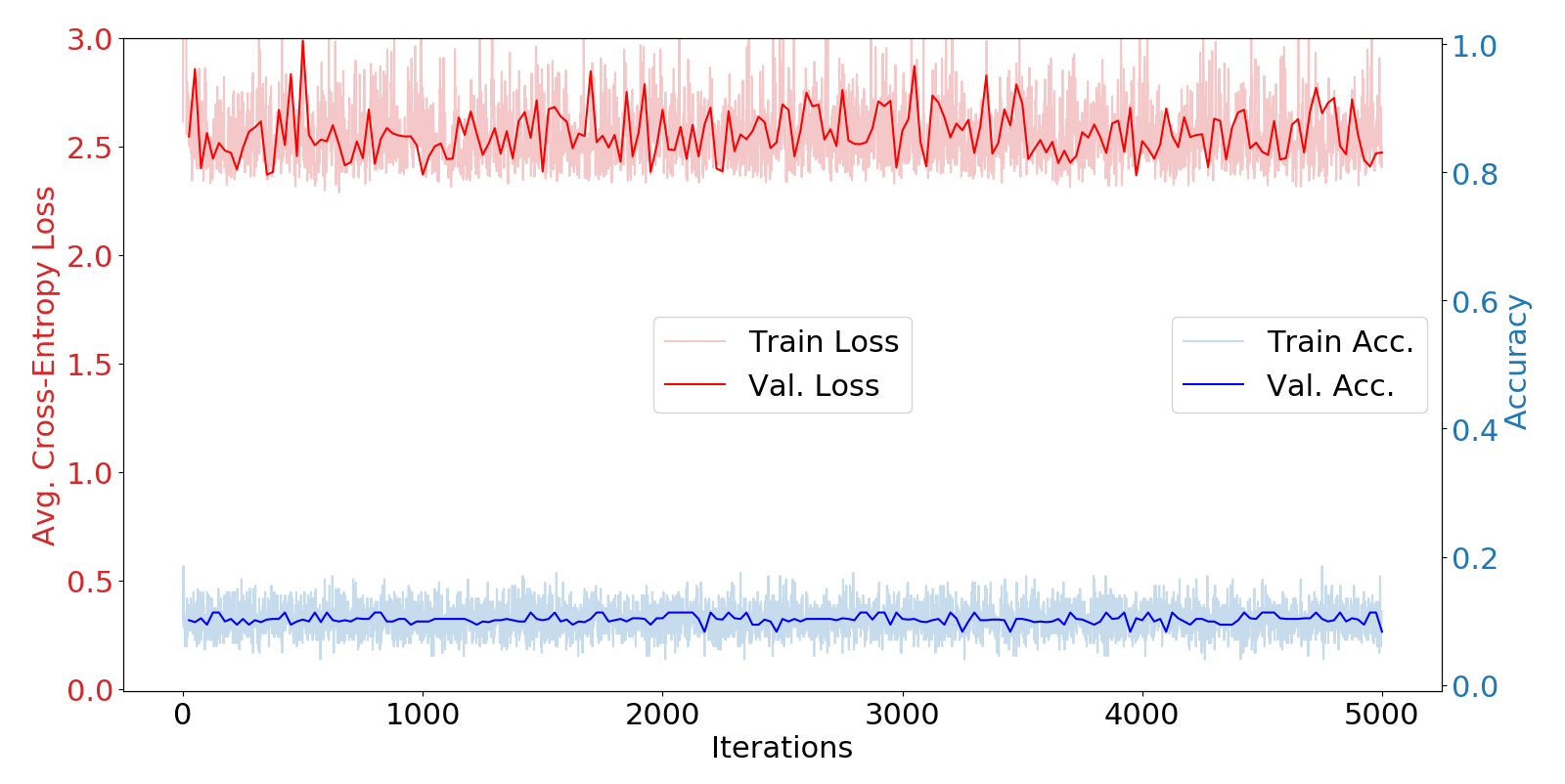
self.grad\_bias = np.sum(grad, axis=0) # = dL/db

grad\_input = grad @ self.weights.T # = dL/dX

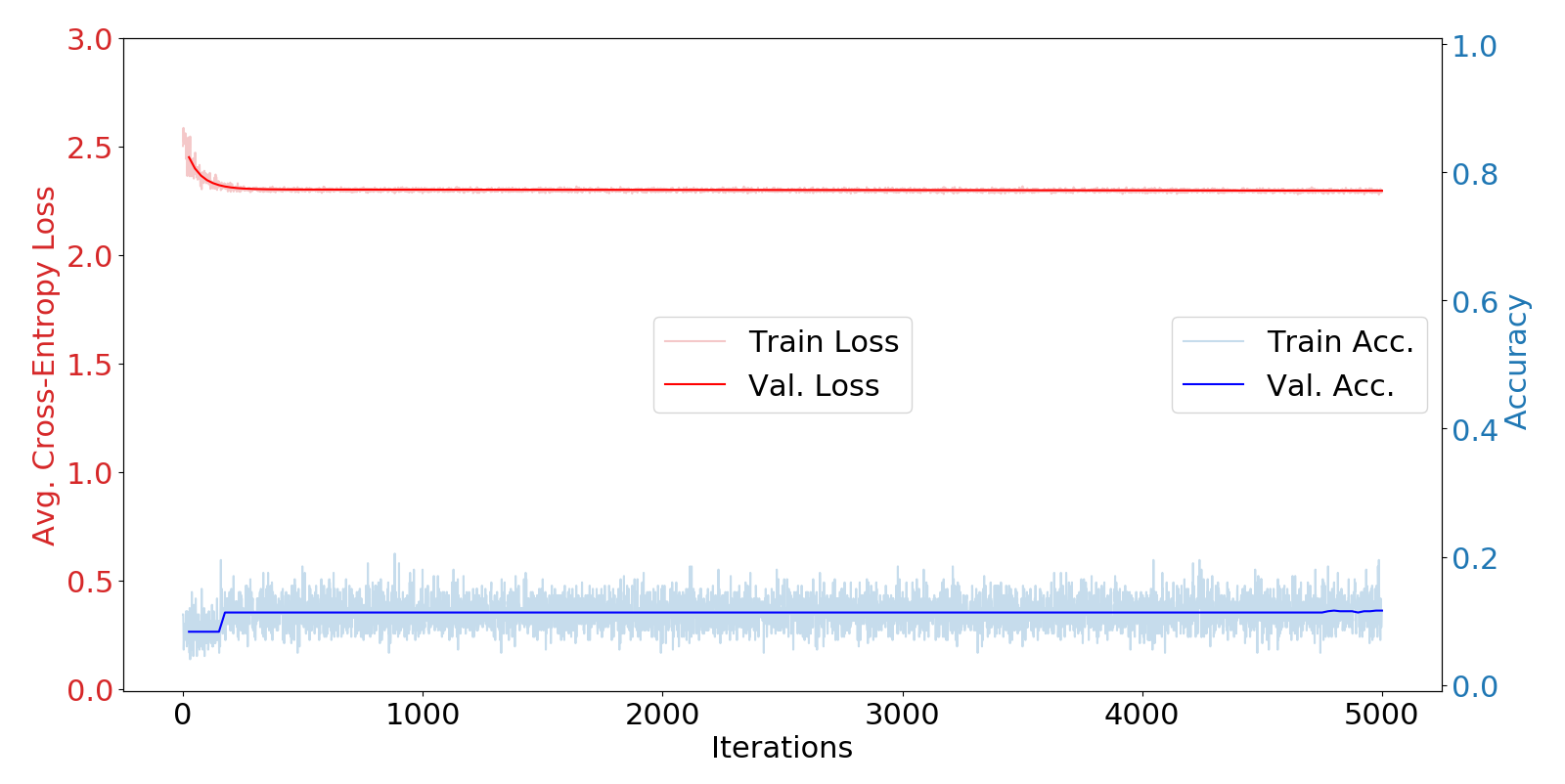
return grad\_input

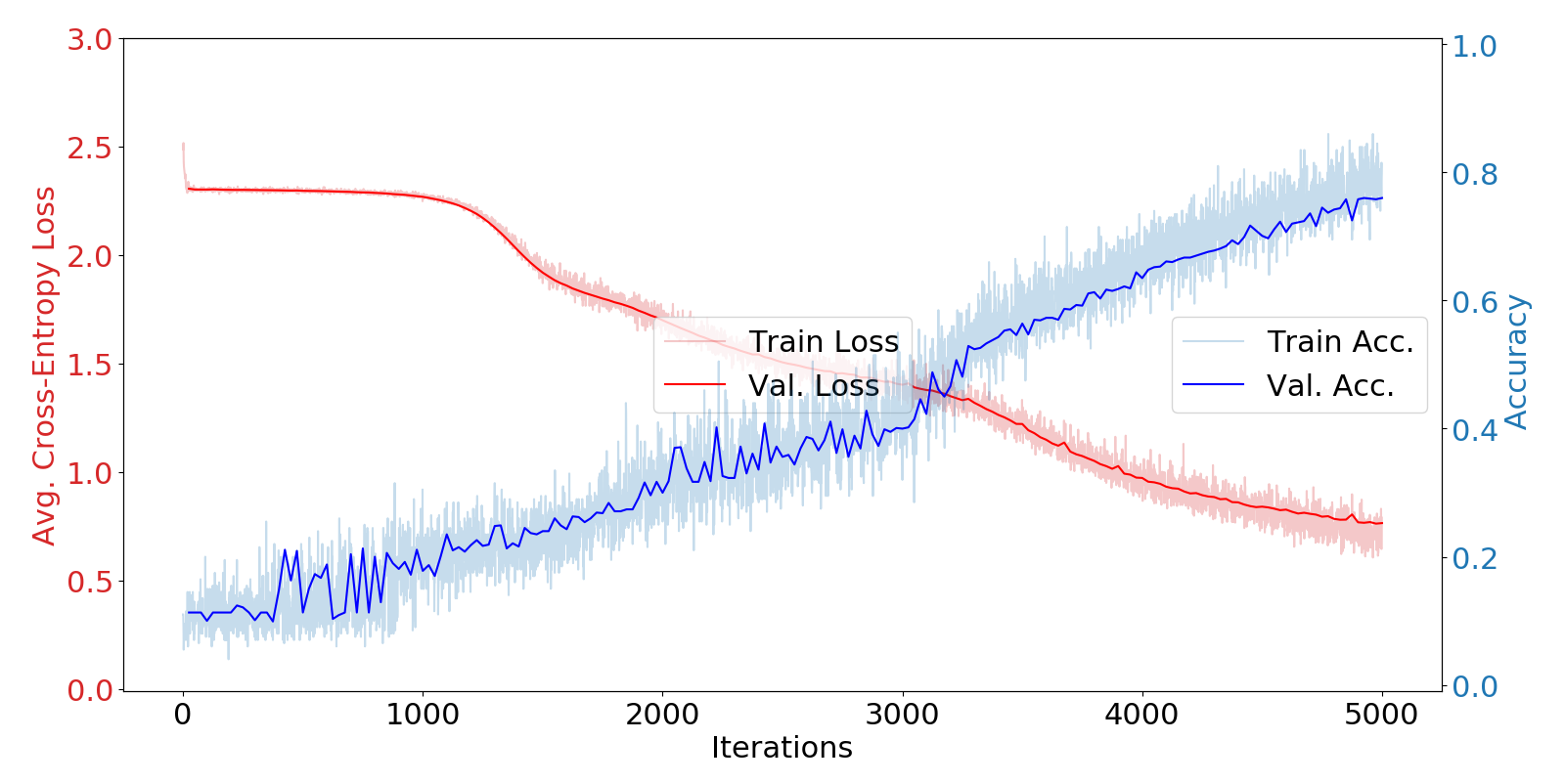
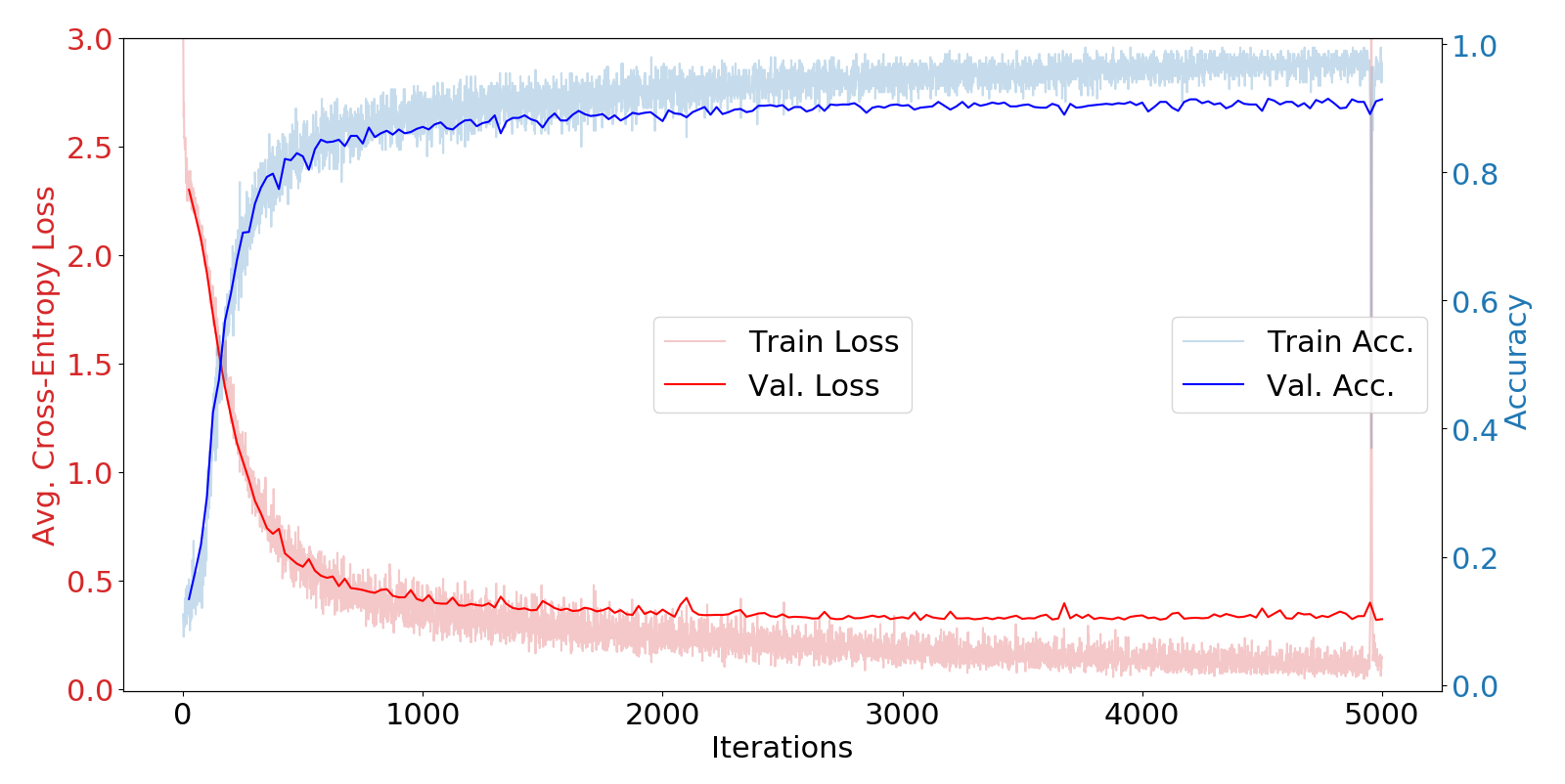
1. 1. S=0.0001 is smooth and slow to converge. S=5 converges immediately and then parts which means major overfitting. S=10 does not converge which means major underfitting.
      1. Step = 0.0001



* + 1. Step = 5
    2. Step = 10
  1. S=0.0001 would eventually converge, S=5 and S=10 would continue to fail.

1. 1. [1] does not converge. [2] converges but its val acc is very erratic. [3] converges quickly and is very stable.
   2. [2] takes larger steps so it's able to test more points. [1] is trending to converge \*eventually\*.
   3. ReLU is just better and uses a constant which helps learn faster in the same number of steps.
      1. 5-layer with Sigmoid Activation



* + 1. 5-layer with Sigmoid Activation with 0.1 step size
    2. 5-layer with ReLU Activation

1. Seed = {1, 10, 100, 1000, 10000}

Val Acc = {85.9%, 88.8%, 88.4%, 89.8%, 87.9%}

All within the 80's with slight variance, it's possible to find a seed with very high or very low validation accuracy.

1. Kaggle -
2. 8 hours
3. Moderate
4. Help with math
5. 60%