It is useful to take note of the time it takes the algorithm to train. This will provide you with an additional level of understanding. That is also your first task (optional).

Using the code from the lecture as the basis, fiddle with the hyperparameters of the algorithm.

1. The \*width\* (the hidden layer size) of the algorithm. Try a hidden layer size of 200. How does the validation accuracy of the model change? What about the time it took the algorithm to train? Can you find a hidden layer size that does better?

2. The \*depth\* of the algorithm. Add another hidden layer to the algorithm. This is an extremely important exercise! How does the validation accuracy change? What about the time it took the algorithm to train? Hint: Be careful with the shapes of the weights and the biases.

3. The \*width and depth\* of the algorithm. Add as many additional layers as you need to reach 5 hidden layers. Moreover, adjust the width of the algorithm as you find suitable. How does the validation accuracy change? What about the time it took the algorithm to train?

4. Fiddle with the activation functions. Try applying sigmoid transformation to both layers. The sigmoid activation is given by the method: tf.nn.sigmoid()

5. Fiddle with the activation functions. Try applying a ReLu to the first hidden layer and tanh to the second one. The tanh activation is given by the method: tf.nn.tanh()

6. Adjust the batch size. Try a batch size of 1000. How does the required time change? What about the accuracy?

7. Adjust the batch size. Try a batch size of 1. That's the SGD. How do the time and accuracy change? Is the result coherent with the theory?

8. Adjust the learning rate. Try a value of 0.0001. Does it make a difference?

9. Adjust the learning rate. Try a value of 0.02. Does it make a difference?

10. Combine all the methods above and try to reach a validation accuracy of 98.5+ percent.