Q1.

(a) $\sum_{k=1}^{m-1} n_k n_{k+1}$

(b) $g(M_1 x)$; $g\left(M_{m-1}\left(\ldots g\left(M_2 g(M_1 x)\right)\right)\right)$

Q2.

(a) Two generic loss functions are cross entropy and MSE. We cannot use error rate since the gradient of error rate vanishes at the last layer already.
The main idea behind SGD is to apply gradient descents on mini batches of training data and propagate backwards through the hidden layers to adjust the parameters.

(b) The benefit of SGD is that it can improve the model using simple calculus while costs less time in comparison to GD.
An epoch is a complete cycle of training the neural network with all the training data, including both feedforward and backprop.
$E\ [T/m]$

(c) Learning rate is the hyperparameter that we can tune to help the model improve. We can start off with a relatively large value then slowly decrease the value to carefully reach the minimum loss. If the rate is too big, the model might end up with even larger loss value, as the movement of gradient gets too reckless. If the rate is too small, the model will see only minimal improvements, which makes the optimization process very time consuming.

Q3. Table 1

id	# of hidden layers	# of neurons in hidden layers	# of epochs	mini batch size	test accuracy	CPU time
1	1	30	30	10	72.19%	1m13.166s
2	1	30	30	50	54.67%	1m2.718s
3	1	30	100	10	76.19%	2m58.267s
4	1	60	30	10	74.48%	1m24.433s
5	2	30, 30	30	10	73.90%	1m2.783s
6	2	40, 20	30	10	73.81%	1m10.923s
7	3	20, 20, 20	30	10	71.24%	0m57.665s

Q4. Table 2

id	# of hidden layers	# of neurons in hidden layers	# of epochs	mini batch size	test accuracy	CPU time
1	1	30	30	10	70.67%	0m58.060s
2	1	30	30	50	42.38%	0m56.388s
3	1	30	100	10	78.57%	3m13.108s
4	1	60	30	10	63.42%	1m29.257s
5	2	30, 30	30	10	65.52%	1m10.546s
6	2	40, 20	30	10	73.52%	1m18.136s
7	3	20, 20, 20	30	10	66.38%	1m7.365s

I changed the code for tanh and tanh_prime functions in network.py.

Q5.

(a) The initial weights and biases are randomly assigned in the initialization function (line 36, 37), so the result of feedforward, which is also the start basis of backprop, is based on random numbers. In addition, the mini batches are randomly selected from the training data in the SGD function (line 63), so that SGD is operated on random batches each time when we run the code.
I think it is desirable and essential for SGD to have randomization for the mini batches to ensure that the model is trained on different distribution of sets each time so that we can eventually achieve a best-performing model. In general, this is more time efficient.

(b) Increasing the batch sizes decreases the test accuracy, while increasing the number of epochs increases the test accuracy. After running the NNs multiple times, I found that the variability in the result increases with less number of epochs. As a tradeoff to higher test accuracy in models with more epochs, it takes much longer to train.