Question 2:

For this question, I choose ClippedGradientDescentOptimizer and tanh for all models.

(b)
For the first RNN model (with one hidden layer):
Relu gate does the best job. Both train loss and validation loss gained by Relu gate are smaller than that gained by Tanh gate. Therefore, training and validation performance are monotonically related.

(d)
For the LSTM model:
Tanh gate gets really nice result. Both train loss and validation loss gained by Tanh gate are very small. However, Relu gate gets totally terrible results. Its train loss and validation loss diverge at EPOCH 2. Before divergence, its train loss and validation loss are really big. Therefore, training and validation performance are monotonically related.

(f)
For the second RNN model (with two hidden layers)
Relu gate does the best job. Both train loss and validation loss gained by Relu gate are smaller than that gained by Tanh gate. Therefore, training and validation performance are monotonically related.

(g)
The final results of three models with two different gates are as follows:

RNN2_relu	127540	1.22923	1.41162	7796.12
LSTM_tanh	127540	1.30444	1.42331	5923.94
RNN1_relu	127540	1.32745	1.45055	3299.29
RNN2_tanh	127540	1.38569	1.51591	7793.22
RNN1_tanh	127540	1.56507	1.64804	3303.34
LSTM_relu	127540	nan	nan	5973.48

It’s easy to find that RNN model with 2 hidden layers with relu gate gets the best result when compared with the other 5 models, and

LSTM with relu gate gets the worst result because its train loss and validation loss diverge at EPOCH 2. We can also find the model with smaller train loss also gets smaller validation loss, so the training and validation performance are monotonically related. Another interesting phenomenon is that relu gate can make RNN model better, while it's a disaster for LSTM model, therefore, relu gate is a better choice for RNN model when compared to tanh gate, however, tanh gate is better for LSTM model when compared to relu gate.