For CNN,I use CnnTextClassifier to be the model of the CNN with the parameter: vocab_size and num_classes. I used Word2Vec vectors of size 500 as input. Then I used the nn.Conv2d with the nn.ModuleList to implement the CNN with the forward step that applies a convolution add max pool layer for each window size. For LSTM, I useTextClassifierLSTM to be the model of the LSTM with the parameter: vocab_size, num_classes, lstm_units, and num_layers. I used Word2Vec vectors of size 500 as input also. Then I used the nn.LSTM to implement the LSTM with the forward step that applies LSTM layers and apply max_pool1d along the time dimension.

Parameters of CNN: (Number of input channels * Filter width * Filter height * Number of output channels) + (Optional bias term * Number of output channels) = (1 *100 * 100 * 10) + (1 * 10)=100010
Parameters of LSTM:2 * 4 * ((Input size * Number of hidden units) + (Number of hidden units * Number of hidden units) + (Optional bias term * Number of hidden units))= 2*4*((100*120)+(120*120)+(1*120))

LSTM have much more training time for one epoch. In CNN, each epoch needs about 1 mins. In LSTM, the runtime of each epoch is decided by the size of batch. Whatever the size of batch is, LSTM need at least 10 mins for one epoch. So that LSTM is more time-consuming.

Changing the activation function would influence the learning process and the performance of the model in different ways that we don't know. For example, for the LSTM model, if we change the activate function in last layer from softmax to relu, the loss would be much larger than using the softmax. It also cause exploding gradients in here. Also changing the activation function would make vanishing and exploding gradients which we would know what happen like ReLU. Otherwise, some of the activation function would reduce the overfitting like ReLU with the large parameters.