1.3.1

For the last label

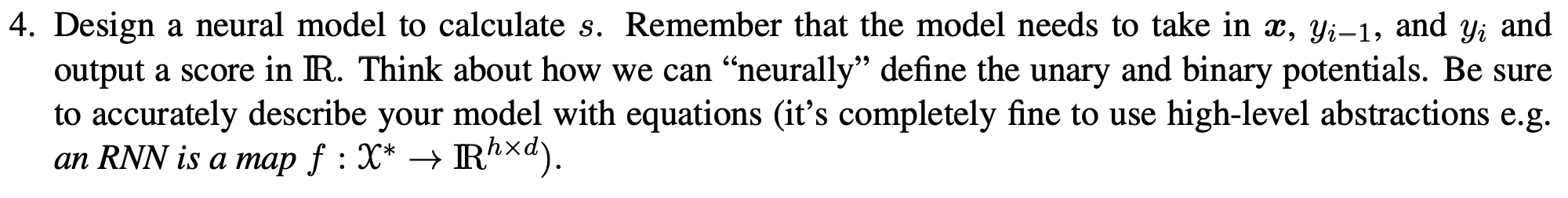
1.3.2

Using dynamic programming to solve this issue. Use a 2d array to keep track of the result of for all possible ys at all possible positions. Also store back pointers along the way.

1.3.3

This algorithm is just a dynamic programming algorithm with time complexity and space complexity . The time complexity is so because for every spot in the 2d array, we need to spend to calculate the maximum score and we have spots in the array. And the array takes space so the space complexity is

1.3.4

Firstly embed the words, then feed words through a bidirectional LSTM encoder, finally pass it through a linear layer to get the unary potential with size batch \* max sequence length \* num\_tags. The binary potential of size num\_tags \* num\_tags is put in a 2d tensor so that it is trainable.

2.2.2

I added elmo embedding to the model. Because elmo can improve just about everything according to lecture.

2.2.4

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Lr | embedding | glove | hiddensize | layers | batchsize | elmo | Accuracy |
|  | 0.01 | 768 | no | 200 | 4 | 64 | no | 64% |
|  | 0.001 | 300 | yes | 64 | 3 | 32 | no | 78% |
|  | 0.001 | 300 | Yes | 300 | 3 | 64 | Yes | 75% |
|  | 0.01 | 300 | Yes | 300 | 2 | 64 | Yes | 92% |
|  | 0.01 | 300 | googleNews | 256 | 3 | 64 | yes | 96% |

3.2

No modification

4. I think the reason I got 90% + is because I used the pre-trained embedding alongside with the elmo embeddings.