1.3.1

$$\heartsuit_j(y_j) = \max_{y_1,\dots y_{j-1}} \sum_{i=1}^{j} s(x,i,y_{i-1},y_i) = \max_{y_1,\dots y_{j-2}} \sum_{i=1}^{j-1} s(x,i,y_{i-1},y_i) + \max_{y_{j-1}} s(x,j-1,y_{j-1},y_j)$$

$$= \max_{y_{j-1}} \left(s(x,j-1,y_{j-1},y_j) + \heartsuit_{j-1}(y_{j-1}) \right)$$

$$\heartsuit_1(y_1) = s(x,i,START,y_1)$$

For the last label y_n

$$\heartsuit_n(y_n) = \max_{y_{n-1}} \left(s(x,n-1,y_{n-1},y_n) + \heartsuit_{n-1}(y_{n-1}) \right) + s(x,n,y_n,STOP)$$

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 $O(\mathcal{L}^2\ell)$ and space complexity $O(\mathcal{L}\ell)$. The time complexity is so because for every spot in the 2d array, we need to spend $O(\mathcal{L})$ to calculate the maximum score and we have $O(\mathcal{L}\ell)$ spots in the array. And the array takes $\mathcal{L} * \ell$ space so the space complexity is $O(\mathcal{L}\ell)$

1.3.4

4. Design a neural model to calculate s. Remember that the model needs to take in $\boldsymbol{x}$, y_{i-1}, and y_i and output a score in $\mathbb{R}$. Think about how we can "neurally" define the unary and binary potentials. Be sure to accurately describe your model with equations (it's completely fine to use high-level abstractions e.g. *an RNN is a map* $f : \mathcal{X}^* \to \mathbb{R}^{h \times d}$).

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.