A Appendix

A.1 Implementation Details

Our models are implemented in Tensorflow and extended from those used in **?**. Our BPE vocabulary was generated with a budget of 10,000. We are using GloVe pretrained word embeddings with 100 dimensions. We used a learning rate of 0.0005, and used the Adam algorithm for optimization. We tuned the LSTM cell size with values of 100 to 300. We tuned the dropout of the network with values of 0.25 to 0.75. We trained the models using the training set, until we stopped seeing improvement on the development set. We then took the best models on the development set to get our test set results.

A.2 MedMentions Details

We restricted the data to 19 different entity types, and documents were split into train, dev, and test splits. The training dataset was then divided into two sets, each containing half of the label types. In each split, entities labeled with a type from the other split are replaced with the 'O' (outside) label. The first set contains 10 entity types, 1,324 documents, and 29,532 entities. The second set contains 9 entity types, 1,309 documents, and 29,466 entities. We trained our models on these two sets, treating them as two different training sets as we did in our other experiments, and tested on the original MedMentions testing dataset.

A.3 Data Stats

Dataset	Docs	Chem	Prot	Disease
CDR	1,500	15,913	0	13,075
-train	500	5,197	0	4,269
-dev	500	5,346	0	4,329
-test	500	5,370	0	4,477
CP	2,432	31,831	30,316	0
-train	1,020	13,017	12,735	0
-dev	612	8,004	7,563	0
-test	800	10,810	10,018	0
WLD	17,381	154,665	14,680	148,971

Table 1: Data statistics for the the different datasets showing the number of annotations for chemical, protein, and disease entities.

A.4 Merging Algorithm

To merge the output of two CRFs from the multi-CRF model, we take the best labels for a sequence from each CRF as given by the Viterbi algorithm. We then merge these predicted labels at the entity level. We go through each prediction of the sequence at the token level, keeping track of the beginning of the current entity. If there is a conflict between the labels, we continue through the sequence until the conflict can be resolved (i.e., both predictions are no longer in their conflicting entities). To resolve a conflict, we favor the CRF that predicted an entity. If both CRFs predicted entities, we favor the CRF that was more confident in its predictions. This confidence is the sum of each token's marginal probability computed over the maximum conflict span. The conflict resolution procedure is detailed in Algorithm 1.

Algorithm 1 Merging entity predictions

A, tokenwise Viterbi predictions from CRF A
$P(A)$, tokenwise log marginal probabilities from CRF A
B, tokenwise Viterbi predictions from CRF B
$P(B)$, tokenwise log marginal probabilities from CRF B
$conflicts$, list of indexes of conflicts between CRF A and CRF B
$output \leftarrow A$
for $start, end$ in $conflicts$ **do**
  if isOutside($A[start : end]$) **then**
    $output[start : end] \leftarrow B[start : end]$
  else if isOutside($B[start : end]$) **then**
    $output[start : end] \leftarrow A[start : end]$
  else if sum($P(A)[start : end]$) $>$ sum($P(B)[start : end]$) **then**
    $output[start : end] \leftarrow A[start : end]$
  else if sum($P(A)[start : end]$) $<$ sum($P(B)[start : end]$) **then**
    $output[start : end] \leftarrow B[start : end]$
  end if
end for
return $output$