# NLU Assignment 2 Report

Lokesh (14355)
lokeshn@iisc.ac.in

## Abstract

Below presented is the report of the assignment 3 which is about Named Entity Recognition using sequence models. Named Entity Recognition is about assigning a tag to each token from a set of predefined tags. This is also a supervised learning problem. The tags that are defined for this assignment are
1. Disease
2. Treatment
3. Others
NER should be designed as a sequence model because the word senses are **ambiguous** and the neighboring words help a lot in the disambiguation process. With the assumption that the proper sense of th word can be resolved by context words we can train a sequence model to achieve a greater accuracy.

## 1 Why do we need context for NER

The context words do provide a great deal of information in tag assignment process. This claim is more evident in entity identification like location. For eg. the words that follow **at, from, etc. are more likely to be location**. This kind of associations are well captured by a sequence model rather than a naive non-sequence model which does nothing more than a naive dictionary lookup to assign tags.

## 2 Conditional Random Fields

**CRF** is the discriminative equivalent of the Generative **HMM** model. Therefore it is clear that through CRF we are only trying to model the probability distribution **P(X/Y)**. In the Graphical model the edges are undirected for CRFs and

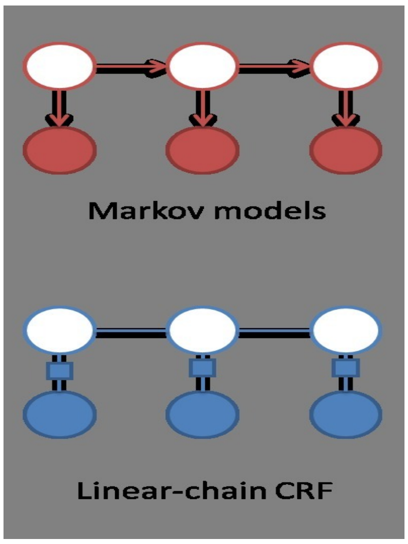

Figure 1: HMM and CRF relation

each node in the graph denotes a random variable. Basically we call it a random field because the state of each node(Random Variable) depends on all the neighbors in the graph. It is through this manner we incorporate context in tag prediction. Hence the tag we predict for the token tag($t_k$) is dependent on features of $t_k$ namely $tag(t_k), isUpper(t_k), isDigit(t_k) etc.$ and also on the tags of the previous tokens namely tag($t_{k-1}$), tag($t_{k-2}$) etc.

## 3 Intuition behind choosing CRF over HMM

Tagging is basically a classification task. In a broader sense we are trying to classify each token to be one among the predefined tag set. Using the learnt model, we would subject each token to the model and which ever tag gets the more likelihood is the predicted tag. It is *empirically* believed that for a classification task, discriminative models have a tendency to outperform the generative models. This is because of the constraint we have on the generative models that they should be able to synthesize the patterns from the underlying data distribution. Hence whatever they do is more or less limited to the raw data. But in a discriminative model because w have no such constraints we can generate as much features as we want from the data *(derived data)* and then approximate the model on top of those features. Also in HMM we assume a conditional independence to make the problem tractable whereas we donot make such independence assumptions in CRFs and this is one other reason why I think CRF would perform better than HMM.

## 4 python-crfsuite

For this assignment I have used python implementation of CRFs. It also accepts the input like mallet ad trains the CRF model and saves all the model parameters which can then loaded again and tested against a test file. Given that the CRF implementation is available handy we will need to design the features which might be appropriate for the task.

## 5 Features Used

I am using the following features for the NER task.

| **unknown words** | **Explanation** |
|---|---|
| word.lower | If all the characters in the word are lower case |
| word.isupper | If all the characters in the word are upper case |
| word.istitle | If the word is a title word |
| word.isdigit | If all the characters are digits |
| word.pos | POS tag for the word |
| word.special | If there are any special characters in the word |
| word.firstcap | If the first character is capitalized |
| word.prefix | First two characters of the word |
| word.suffix | Last two characters of the word |
| word.ortho | Orthographic feature which encodes the structure for the word Xxd% X stands for capital letter, x for small letter, d for digit, % for special character |

Table 1: Features for CRF

## 6 Comment on Analysis

The analysis on the output generated by the code is illustrated in the form of graphs. I have calculated the following measures on the results
1. Precision => False Positives
2. Recall => False Negatives
3. F1 Measure => Tradeoff between the two

All the measures are reported only for the labels (D, T) only. Because the data is so much skewed in favor of the label O, the label O dominates the rest. Hence if we blindly assign the tag as ’O’ for all the tokens **we get accuracy around 80% for free**. For all the analysis the model is truncated at 50 iterations.

## 7 Cross Validation Analysis

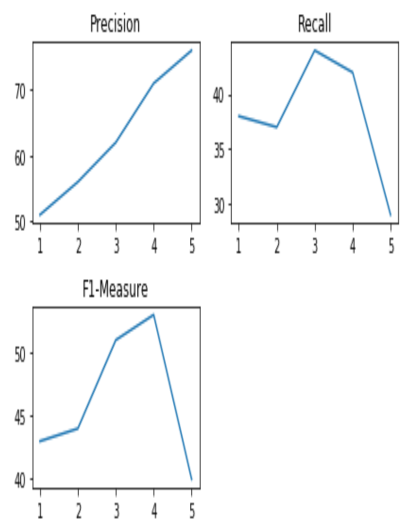

Figure 2: Cross Validation Analysis

As we can see from the graph that precision and recall are on average around **40%**. This is primarily because of the skew in the labels in the input data. Empirically we can observe that for this particular split the data split number 4 happened to give good results. Multiple runs of the code didnot produce similar kind of graph primarily because of the randomness in the data splits.

## 8 Why is Precision and Recall less

This argument is adapted from the following paper Using Non-Local Features to Improve Named Entity Recognition Recall. They have mentioned in the paper that using just local features the best model can possibly achieve around 25% precision and recall. They have also identified the reason as data skew in favor of Others label. They have claimed that unless we model non-local features (long distance dependencies) explicitly CRFs cannot do better. Usually CRFs have a high precision and low recall because they overproduce the O label.

## 9 Effect of number of Iterations

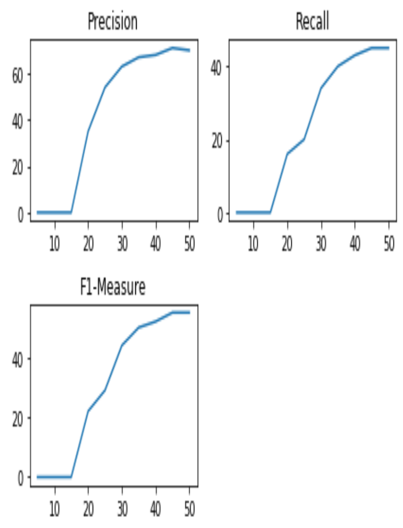

Figure 3: Effect of number of Iterations

As we can see from the above graph, the accuracy keeps getting better as the number of iterations for which we run the model increases. Particularly what is intrseting is that for the initial 15 runs the algorithm doesnt learn anything and simply outputs all the tokens as others. However after 15 iterations w can observe that all the three measures become better and better.

## 10 Effect of each individual feature on the model

Basically we need to perform an ablation study over the features we have considered. Hence, *to assess how good each feature is we can measure how bad the model would perform in the presence of every feature other than that feature*. Presented below are the graphs for each of the ten features we have considered. Below presented are the consolidated results where a group of features are removed and accuracy is measures with 5-fold cross validation.

| **Cross Validation Number** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| **Precision** | 52 | 49 | 59 | 70 | 76 |
| **Recall** | 38 | 37 | 44 | 45 | 26 |
| **F-1 Measure** | 43 | 42 | 50 | 55 | 41 |

Table 2: Features Removed : lower, upper, title, digit, special, firstcap

| **Cross Validation Number** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| **Precision** | 56 | 56 | 62 | 74 | 75 |
| **Recall** | 28 | 36 | 40 | 41 | 24 |
| **F-1 Measure** | 36 | 43 | 48 | 53 | 35 |

Table 3: Features Removed : POS, Prefix, Suffix, Ortho

Eventhough the results from the first table are looking better than that in the second one, there is no explanation as to why it is happening. The difference is not significant and one can only attribute this observation to the data we have.

## 11 Conclusion

I conclude this assignment by saying that as long as we dont exploit the non local properties with long distance dependencies the recall measur and hence the F-1 measure would not improve much. Therefore it is ubiquitous that Neural models like LSTM which model long distance dependencies might do better.

## 12 References

1. Mallet Lecture Notes
2. CRF Tutorial

3. Py-CRF 3. Using Non-Local Features to Improve Named Entity Recognition