CS571 Project2 ReadMe and Result Discussion

Author: Jingzhi (John) Wang Email: jwang67@emory.edu

Where-to-find
Runnable jar files under /home/jwang67/cs571/P2NER/CS571ProjectTwo/dist/
Source code under /home/jwang67/cs571/P2NER/CS571ProjectTwo/src/cc/assignment/

How-to-run
Same for task1.jar, task2.jar, task3.jar
Command Line Input : java –jar task1.jar [trainDir] [testDir]
args[0] [trainDir] where training data is provided
Default is /aut/proj/ir/eugene/Data/CS571/project2/data/train , if you don't enter anything
args[1] [testDir] where testing data is provided
Default is /aut/proj/ir/eugene/Data/CS571/project2/data/test , if you don't enter anything

Task1 Baseline

Basic idea:
1. Count each label for certain words
2. Map label and counts to the word in wordFeatures
3. Iterate through training data
4. Test the testing data and get the most probable label
5. Evaluate and report Precision/Recall/F1 value

Result(Abridged):

Label	Precision	Recall	F1
B-Peop	0.769	0.743	0.756
I-Peop	0.551	0.321	0.406
B-Loc	0.667	0.333	0.444
I-Loc	0.250	0.200	0.222
B-Org	0.333	0.500	0.400
I-Org	0.625	0.278	0.385
B-OrgCorp	0.586	0.773	0.667
I-OrgCorp	0.625	0.588	0.606
B-OrgPolBody	0.714	0.714	0.714
I-OrgPolBody	0.667	0.667	0.667
B-OrgTeam	0.905	0.905	0.905
B-OrgUniv	1.000	0.600	0.750
I-OrgUniv	1.000	0.500	0.667
Overall	0.807	0.748	0.777

More detailed result can be obtained by running the task1.jar.

The overall performance of the result is reasonable at 77.7% for F1 measure. And it seems that more specific category achieve a higher F1 measure. Meanwhile, "Beginning" label ("B-")

tends to outperform “In” label (“I-”).

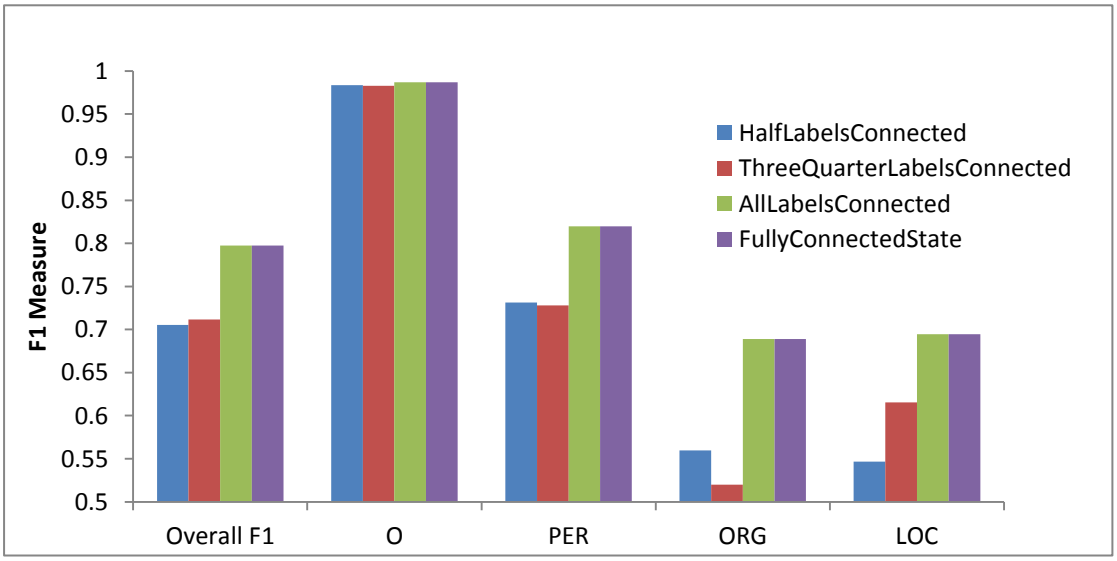
Task2 CRF for Major NER Label

Basic idea:

1. Constructed a pipe called “ImprovedSimpleLabelNERdata2TokenSequence” to extract features such as label/previousWord/nextWord/POStag/phrase in the input data
2. Push input data into the list of pipes and convert sub-category labels into O/PER/ORG/LOC
3. Use built-in CRF trainer in MALLET package to find most Likelihood label over test data
4. Iterate to find most optimal weights for each feature
5. Evaluate and report Precision/Recall/F1 value on each iterate

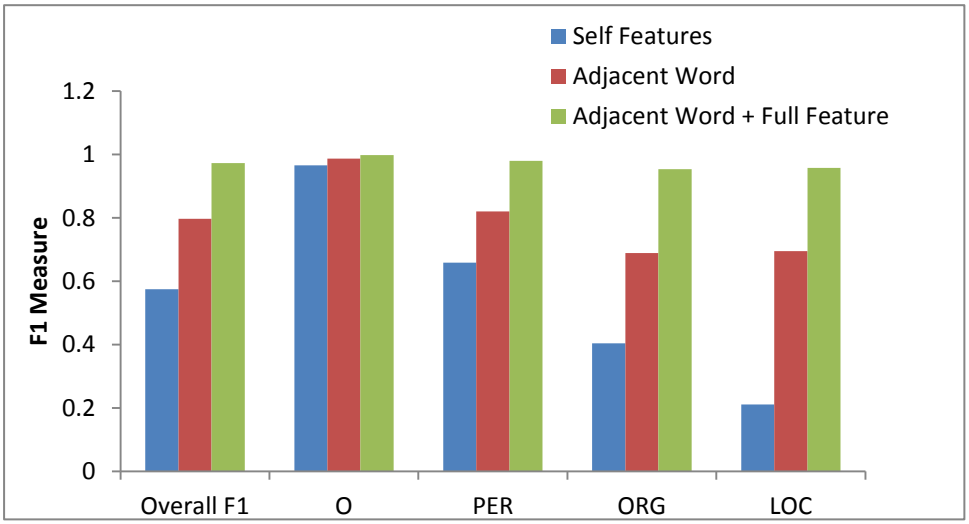
Result

1. Testing State Connections



With only 4 states (O/PER/ORG/LOC), the transition from state to state is a limited number. Thus, removing a portion of label connections will compromise the result greatly. Under CRF training, NER task can achieve upto 79.8% F1 measure for simple labels (taking only the adjacent word into consideration).

2. Feature inclusion



In Self Features, the feature extraction step only takes the label/ POStag/ phrase features of the word itself into consideration. In Adjacent Word, the feature extraction step takes adjacent word order into account. In Adjacent Word + Full Feature, the feature extraction step takes all features for the adjacent words including label/ POStag/ phrase features plus the word order.

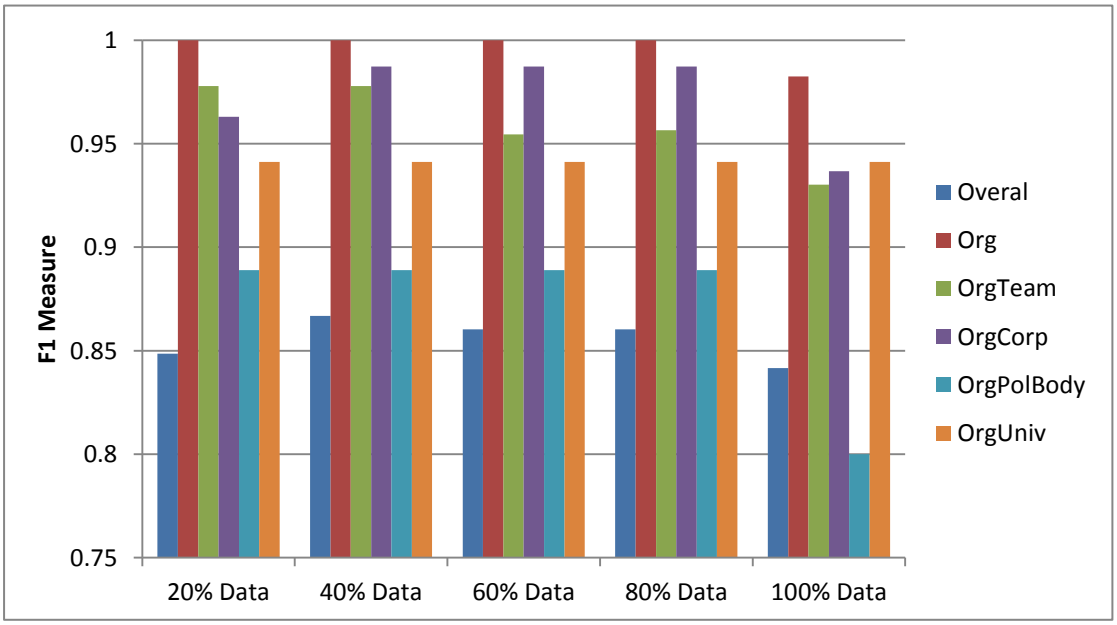
The result indicates, well of course, more features will lead to better results. But the efficiency was compromised. Overall F1 measures for each trial are 0.5751, 0.7976, 0.9727 which, in some sense, can match up to the state-of-art methods.

Task3 CRF for Sub-labels in NER

Basic idea:

1. Similar to task2.
2. Constructed a pipe called "ImprovedDetailedLabelNERdata2TokenSequence" to extract features such as label/previousWord/nextWord/POStag/phrase in the input data
3. Maintain the subcategory for Org labels while converting all other labels to general categories

Result



Overall, the training is much more expensive to proceed. It seems there is an over-fitting problem for sub-category labels. The reason might be the too much weight is added and CRF trainer algorithm cannot reasonably resolve and optimize likelihood.