Homework 7

Authors: Srinivas Suri, Ilayda Ipek Onur

Part 1:

We tried adding 12 features:

0. The word itself as a feature
1. Suffix of length 3
2. Suffix of length 4
3. Prefix of length 3
4. Whether first letter is capitalized
5. Whether the word is fully capitalized
6. Whether the word is fully lowercase
7. Whether the word contains a hyphen
8. Whether the word contains an apostrophe
9. Length of the word
10. Word shape of the word
11. Get the pos tag from the nltk lib

We observed that pos tag from the nltk lib, prefixes and suffixes didn't improve our accuracy. Therefore we just kept the features that are colored in green.

We also experimented with window sizes [-1,1], [-2,2], [-5,5] and [-7,7].
Window size [-5,5] gave the best accuracy.We present the f1-scores we achieved with window size [-5,5] in the below table:

Features	**F-1 Scores**
0,4,5,6	69.09
0,4,5,6,7,8	68.97
0,4,5,6,7,8,9,10	68.67
0,4,5,6,7,8,9,1,2,3	67.50

Adding few basic features like if the word consists of only lower characters, if the word consists of only upper case letters and if the first letter is capitalized proved to be powerful features and results in 69.09% dev set f1-score. Adding features 7,8,9,10 also improved the f1-score on the

test set to 73% that they are also strong indicators on the NER's and this is what we used in the submitted model.

Our intuition behind adding these features is that. Often, Capitalised first letters, fully upper case words hint at a word belonging to a NER rather than belonging to other categories. Hence, we added these features and it turned out to be very useful for the classification task.We next experimented with hyphen and aphostrophe but they didn't increase the performance by a huge margin.

Part 2:

We have experimented with various classifiers with a window size of [-2,2] and have presented the results below:

MutlinomialNB	train	.93	50.17	49.63	49.90
MutlinomialNB	dev	.9	39.72	38.52	39.11
LinearSVC	train	1	96	97.66	96.83
LinearSVC	dev	.96	65.99	71.32	**68.55**
Perceptron alpha= 0.01	train	.94	41.88	49.39	45.33
Perceptron alpha= 0.01	dev	.92	36.33	42.17	39.03
Random Forests estimators=20	train	1	97.32	98.55	97.98
Random Forests estimators=20	dev	.95	55.57	60.79	58.06
Random Forests estimators=10	train	1	95.7	97.55	96.61
Random Forests estimators=10	dev	.94	51.82	58.77	55.08
DecisionTree	train	1	98.17	99.05	98.6

DecisionTree	dev	.95	52.82	62.05	57.09
Logreg C=1	train	.99	85.82	89.39	87.42
Logreg C=1	dev	.95	64.11	67.87	**65.94**
Logreg C = 0.01	train	.95	54.58	58.92	56.62
Logreg C=0.01	dev	.93	48.95	53.00	50.89

Analysis:

The set of features we used range from features: 4-10 mentioned in the part 1. Our feature set resulted in a data where linear classifiers worked the best. So, Logistic regression and Linear SVC fit the data better than the decision trees/random forests. Although forests and decision tree ove fit the training data. To alleviate this problem, we tried the random forests too, but, linearsvc() and logistic regression classifiers performed better as the data was linearly separable when features: 4-10 were used.

We took the best found classifier from this window , the LinearSVC() and experimented with a window size of [-5,5] and [-7,7]. We achieved a f1-score of 68.7% with window size 10 and 67.77 with a window size of 14. Our best classifier we submitted is LinearSVC() with a window size of 10 ([-5,5])and using features from 4-10 .