CIS 530 Assignment 7: Named Entity Recognition

Ignacio Arranz
Yezheng Li

Problem 1. *What features or resources you used, and how they helped.*

Answer. *The features we used were:*

- *word: e.g. "Carlos"*
- *word shape: e.g. "Xxxxxx"*
- *word short shape: e.g."Xx"*
- *word prefix: first character e.g. "C"*
- *PoS: part of speech*
- *all upper: boolean whether all upper chars*
- *has hyphen: boolean whether it had a hyphen*
- *sentence frequency: used*

Problem 2. *What changes (if any) you made to the learning algorithm.*

Answer. *We changed the model to be a logistic regression. W tried several different algorithms, and logistic regression provided the best results.*

We also modified the length window of words that was analyzed when creating the features for a given - we included 4 words to the left and 4 to the right.

We tried to apply PCA, in order to reduce dimensionality, but this required making the matrix of features dense, which crashed our program. Random forest, AdaBoost, GradientBoost did not produce good results.

Problem 3. *What extra techniques you used (for example, two passes).*

Answer. *We use two-passes to improve our result from 71.26% to 73.01%. Our second pass is to fix the case when*

1. *one word w_i is labeled 'I-ABC',*
2. *but the previous word w_{i-1} not labeled 'B-ABC' or 'I-ABC' (that is, the previous word is labeled 'O', 'I-XYZ' or 'B-XYZ').*

and is implemented like this:

1. *For word w_i, we summarize its count over all labels ('LOC', 'ORG', 'MISC', etc., that means, 'B-LOC' and 'I-LOC' are treated same labels; we do not count 'O') as a dictionary $count[w_i]$;*
2. *if a conflict case (mentioned above) appears, we just modify the second word in the following way: compare w_{i-1}*

$$\frac{\max count[w_i]}{\sum count[w_i]}, \frac{\max count[w_{i-1}]}{\sum count[w_{i-1}]}, \tag{1}$$

   - *if the former is larger, we adjust label of w_i by $\arg\max count[w_i]$;*
   - *otherwise (that is, the latter is larger), we adjust label $I - ABC$ of w_i by label $? - XYZ$ of w_{i-1}, that is, change label of w_i into $I - XYZ$.*

3. *To be honest, finally we obtain 72.51% on test set while we obtain 73.01% with an obvious bug we found in an old version.*

Problem 4 (Optional, but cool)**.** *What papers you read, and how your method compares to them.*

Answer. *The paper we read was [Lample et al., 2016b]. Our method, when compared to them is clearly inferior when comparing performance. This draws our attention to CRF (we feel uncomfortable about LSTM's "overfitting" issue in HW6).*

Model training in [Lample et al., 2016b, Lample et al., 2016a] is very slow (on my Macbook pro 2016, it takes about 8-10 hours).

Problem 5 (If you did the unconstrained)**.** *What code you downloaded, and what parameters you set.*

Answer.
1. *We implemented as black box [Lample et al., 2016b, Lample et al., 2016a]. Model training is very slow (on my Macbook pro 2016, it takes about 12-16 hours):*
   (a) *we just use default parameters;*
   (b) *datasets are downloaded from [Parviainen, 2010].*
   (c) *we only trained up to epoch 51 and stopped (we do not have enough time);*
   (d) *Recognizing that for evaluate function calls of tagger and train procedures in [Lample et al., 2016a] are different, we made the two evaluate function calls to be the same. Meanwhile, we change everything into python3.6. We eventually reaches 79.19%*
2. *We also implement CRF from [Mikhail Korobov, 2017]. It seems only changes the classifier. The original code has F1-score 53%.*

We do some of our feature adjustments (we cannot do all feature adjustments the same way as logistic regression which results in bad performance). Our CRF gets about 57%.

For unconstrained result leaderboard, we submit 79.19% from LSTM-CRF [Lample et al., 2016b, Lample et al., 2016a].

References

[Lample et al., 2016a] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016a). Ner tagger. https://github.com/glample/tagger.

[Lample et al., 2016b] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016b). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

[Mikhail Korobov, 2017] Mikhail Korobov, K. L. (2016-2017). Named entity recognition using sklearn-crfsuite. http://eli5.readthedocs.io/en/latest/tutorials/sklearn_crfsuite.html.

[Parviainen, 2010] Parviainen, T. (2010). My explorations in natural language processing. https://github.com/teropa/nlp/tree/master/resources/corpora/conll2002.