Intermediate Report

Nishant Agarwal, Shweta Singhal

April 3, 2017

1 Project Description

Our project for the course is to create a dependency parser for Hindi and Urdu dataset using Transition based Arc-Standard parsing algorithm[5].

2 Data Description

The dataset is collected from a Hindi-Urdu Treebank Project [1], where the data is available for research and academic purpose. The dataset contain both Hindi and Urdu corpus, available in CoNLL-X and SSF(Shakti Standard Format) format, with utf and wx encodings separately. The wx encoding stands for ROMAN ALPHABETIC CODING SCHEME FOR INDIAN LANGUAGES. We have decided to go with wx encoding for Hindi dataset currently to get a baseline ready.
Our Hindi dataset consists of Training, Testing and Development set distribution with $933, 131$ & 112 documents in each set respectively, with a range of $6 - 30$ sentences in each document and Urdu dataset consists of Training, Testing and Development set distribution with $625, 60$ & 35 documents in each set respectively with a range of $5 - 10$ sentences in each document.

3 Project Status

We have used MaltParser[2] API for its Arc Standard Oracle function. The API allowed us to generate transitions and configurations for each sentences. Each configuration is defined in terms of stack and a buffer, where the stack is initialized with ROOT element and buffer is initialized with all the words in a sentence. The Arc Standard Oracle predicts the most probable transition based on the current configuration. The transitions are used as Gold Standard labels for a given configuration. The transition is applied on the current configuration resulting in a new configuration. This sequence goes on till the buffer becomes empty. Each configuration is a training instance and the label is the transition that we got.
The features are extracted from the configurations as described in Table 1 of Chen and Manning[3] which are a combination of features described in Zhang and Nivre 2011[6] and Huang et. al[4]. The features are binarized later to liblinear format. Overall in our Hindi corpus datatset we have around $533, 432$ unique features from $333, 829$ training instances. The test data gave $42, 204$

test instances. We used the liblinear library and ran multiclass classifiers and measured the accuracy. Below are the results from the classifier:

Classifier	**Accuracy**
L2-regularized logistic regression	83.3712%
L2-regularized L2-loss support vector classification (dual)	82.5372%
L2-regularized L2-loss support vector classification (primal)	83.1556%
L2-regularized L1-loss support vector classification (dual)	82.7315%
support vector classification by Crammer and Singer	82.4116%

Table 1: **Classificaiton Accuracy Table**

4 Future Plan

With the Oracle function learned, our plan of action for the final deadline is:

- Implement all the other features as suggested in Chen and Manning[3].
- Use Greedy inference to recreate dependency graphs.
- If time permits, Beam Search as inference to recreate dependency graphs.
- Use MaltEval to evaluate the dependency graphs.
- Perform evaluation on Urdu dataset as well.

References

[1] Hindi-urdu dependency treebanks (hutb). `http://ltrc.iiit.ac.in/hutb_release/`.

[2] Maltparser. `http://www.maltparser.org/`.

[3] Danqi Chen and Christopher D Manning. A fast and accurate dependency parser using neural networks. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[4] Liang Huang, Wenbin Jiang, and Qun Liu. Bilingually-constrained (monolingual) shift-reduce parsing. In *Proceedings of EMNLP*, 2009.

[5] Sandra Kubler, Ryan McDonald, Joakim Nivre, and Graeme Hirst. *Dependency Parsing*. Morgan and Claypool Publishers, 2009.

[6] Yue Zhang and Joakim Nivre. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers - Volume 2*, HLT '11, pages 188–193, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.