Dor Cohen 301216321
Itai Gat 311148449

NLP HW2 Report - Winter 2018/19

Introduction

We trained a classifier for dependency parsing using structured Perceptron. Two models were optimized namely (1) which is baseline with predefined features, and (2) not restricted.

Model

Both include all features reported in [McDonald 2005], see appendix A for detailed list.

Data and training procedure

Both models were trained on identical train (5000 sentences) and test set (1000 sentences), training was done using edge based factorization with structured Perceptron algorithm. During training no filtering was performed based on features occurrence.

Inference

Method for inference was Chu-Liu-Edmond, without further modifications.

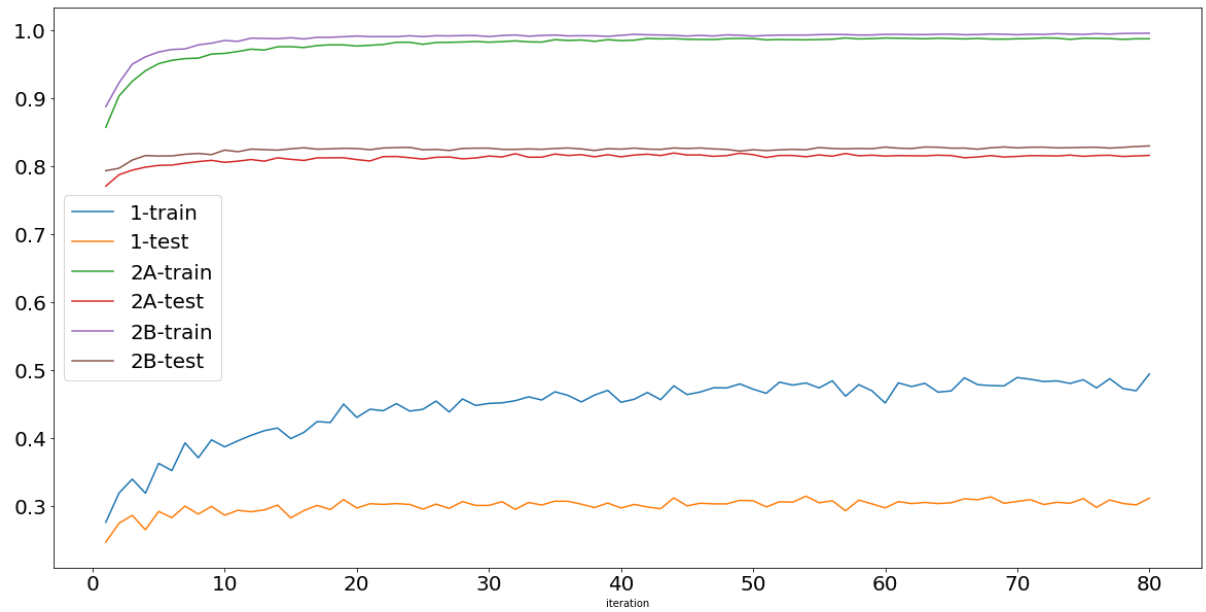
Results

Accuracy (best):

Model	Train	Test
(1) - baseline	48.06 %	31.39 %
(2) - A	98.73 %	81.89 %
(2) - B	99.53 %	**82.93 %**

- Submitted models (1) and (2)-B

Accuracy per training iteration:



Discussion

We experimented with structured Perceptron for the task of classifying a dependency tree from a sentence and its Part-of-speech tagging, we have been provided with a ‘baseline’ model, in terms of the features it includes. Through our trials we have observed how adding a single attribute, namely the distance between parent and child, improves our model results significantly. Moreover, allowing to extract features from a wider window size around the word also contributes to the classification results. Regarding generalization, we expect our best model performance on the tagging task to decrease compared to the test results, due to the large gap between train and test accuracies which suggests on overfitting.

Technical details

Optimization and inference were performed on Ubuntu 16.04 machine with 4GB of RAM. Model (1) and (2) fit procedure takes around 360 seconds per iteration, or 8 hours for 100 iterations, tagging comp takes ~20 seconds. Dependencies: *Python* 3.5.2, *Numpy* 1.16.0

Team work

Training procedure was implemented jointly, Dor implemented features representation, Itai implemented text decoding and inference algorithm, experiments and report were performed and written together.

Appendix A - Feature counts

	Features	Baseline **without d(p,c)**	Model (2) - A	Model (2) - B
1	p-word, p-pos, d(p,c)	9993	28387	same as Model (2) - A
2	p-word, d(p,c)	8876	26440	
3	p-pos, d(p,c)	37	216	
4	c-word, d(p,c)	15908	27505	
5	c-word, d(p,c)	14162	26087	
6	c-pos, d(p,c)	45	301	
7	p-word p-pos, c-word c-pos, d(p,c)	-	78872	
8	p-pos, c-word, c-pos, d(p,c)	31314	40985	
9	p-word, c-word, c-pos, d(p,c)	-	78290	
10	p-word, c-word, c-pos, d(p,c)	33936	46319	
11	p-word, p-pos, c-pos, d(p,c)	-	78518	
12	p-word, p-pos, c-word, d(p,c)	-	77914	
13	p-pos, c-pos, d(p,c)	749	2660	
14	p-pos, np-pos, pc-pos, c-pos, d(p,c)	-	23566	
15	pp-pos, p-pos, pc-pos, c-pos, d(p,c)	-	21475	
16	p-pos, np-pos, c-pos, nc-pos, d(p,c)	-	24450	
17	pp-pos, p-pos, c-pos, nc-pos, d(p,c)	-	25884	
18	p-pos, nnp-pos, ppc-pos, c-pos, d(p,c)	-	-	36340
19	ppp-pos, p-pos, ppc-pos, c-pos, d(p,c)	-	-	33185
20	p-pos, nnp-pos, c-pos, nnc-pos, d(p,c)	-	-	35388
21	ppp-pos, p-pos, c-pos, nnc-pos, d(p,c)	-	-	40141
Total		115020	607829	752883

- Where d(p,c) is the distance between parent and child reduced to the range [-4,4], pp-pos means previous parent pos, and nnp-pos means next next parent pos

Appendix B - Results screenshots

Model (1)

```
total features: 115020
'__init__'  0.27 ms
Accuracy:0.3139979445015416
```

Model (2)

```
total features: 752883
'__init__'  0.94 ms
Accuracy:0.829393627954779
```