Statistical Parsing with Unsupervised Domain Adaptation

Harshal Priyadarshi*

University of Texas at Austin
harshal@cs.utexas.edu

Abstract

I. Introduction

Statistical Parsing is the process of finding the most probable parse of a raw sentence, based on the Probabilistic Grammar learnt on the parsed data, using a count representation of each grammar production.
As we stated there is a problem that the parser can be very specific to a domain. We will see this in the implementation. There is a semi-supervized algorithm that I call "Out-of-domain Pseudo Training" which we show as improving the parse performace over the Out-of-domain data.
In general we can't develop an accurate full parse of a sentence. Thus we need to use a metric to describe the correctness of the parse. We will be using the F1 Score metric for our parse tree, which is based on the Precision and Recall, which I will describe in the later section.

The general flow of the report will be as follows. I will start with the parse Tree correctness metric in Section II. Then will describe the "Out-of-domain Pseudo Training" in section III, and then follow up with the Experiment description in section IV, and then end with the discussion in section V.

II. Parse Correctness Metric

I. Parse Precision

$$Precision = \frac{Number\ of\ correct\ constituents}{Number\ of\ Constituents\ in\ Empirical\ Parse}$$

II. Parse Recall

$$Recall = \frac{Number\ of\ correct\ constituents}{Number\ of\ Constituents\ in\ Correct\ Parse}$$

III. Parse F1 Score

$$F1\ Score = \frac{2(Precision)(Recall)}{Precision+Recall}$$

III. Algorithm

This algorithm is also called Self-training algorithm.

1. Take a part of parsed data, as seed data.
2. Train the Syntactic PCFG model by generating the Probabilitstic Production by just counting.
3. Now take a part of out-of-domain data, as self-training data.
4. Generate a probabilistic parse for the self-training data using the trained parser.
5. Merge the seed data and self-training data together, and then train the parser again.
6. This is the final parser. To test its efficiency, train this new parser and the old parser (just trained on seed) on the remaining part of the out-of-domain data (data not used for self-training, to prevent overfitting).

IV. Experiment

The dataset chosen was the WSJ and Brown data. The following experiments were performed:

- Train on WSJ

*Masters in Computer Science UT Austin, Fall 2015

  - Varying Seed Data
    - Algorithm - Self Training on Brown and Test on Brown
    - Base Case -Without Self Training, Test on Brown
    - In-domain - Without Self Training, Test on WSJ
  - Varying Self Training Data
    - Self Training on Brown and Test on Brown

- Train on Brown
  - Varying Seed Data
    - Algorithm - Self Training on WSJ and Test on WSJ
    - Base Case -Without Self Training, Test on WSJ
    - In-domain - Without Self Training, Test on Brown
  - Varying Self Training Data
    - Self Training on WSJ and Test on WSJ

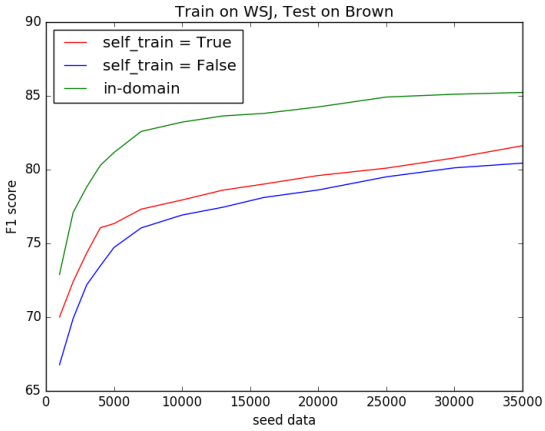
The results obtained are as follows:

I. Seeded on WSJ

Varying Seed Data
The results of the algorithmic, base and in-domain experiments with seed data from WSJ and self training data from Brown are shown below:

Color	1k	2k	3k	4k	5k	7k
Red	70.00	72.40	74.35	76.06	76.34	77.32
Blue	66.77	69.90	72.19	73.48	74.72	76.05
Green	72.89	77.09	78.83	80.29	81.16	82.59

10k	13k	16k	20k	25k	30k	35k
77.94	78.61	79.02	79.60	80.10	80.79	81.62
76.92	77.45	78.11	78.62	79.51	80.12	80.44
83.22	83.64	83.81	84.25	84.92	85.11	85.23



Varying Self Training Data
Now keeping the seed data fixed and varying the Brown self training data we get the following plot.

1k	2k	3k	4k	5k
77.05	77.42	77.73	77.77	77.80

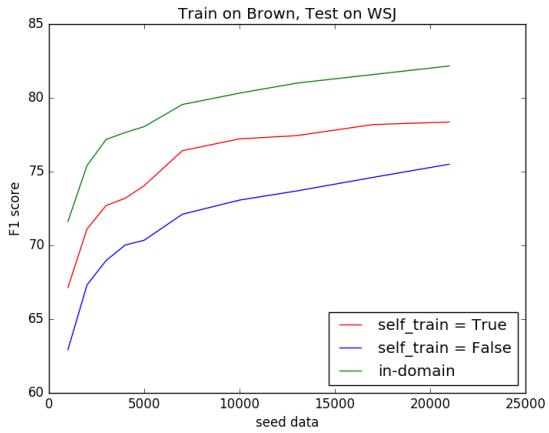
7k	10k	13k	17k	21k
77.88	77.92	77.97	77.93	77.91

II. Seeded on Brown

Varying Seed Data
Doing the similar counterpart for Brown as seed we get:

Color	1k	2k	3k	4k	5k
Red	67.14	71.11	72.70	73.19	74.05
Blue	62.93	67.32	68.97	70.02	70.35
Green	71.62	75.41	77.18	77.65	78.05

7k	10k	13k	17k	21k
76.42	77.22	77.44	78.19	78.36
72.11	73.07	73.69	74.61	75.50
79.55	80.32	81.00	81.58	82.16

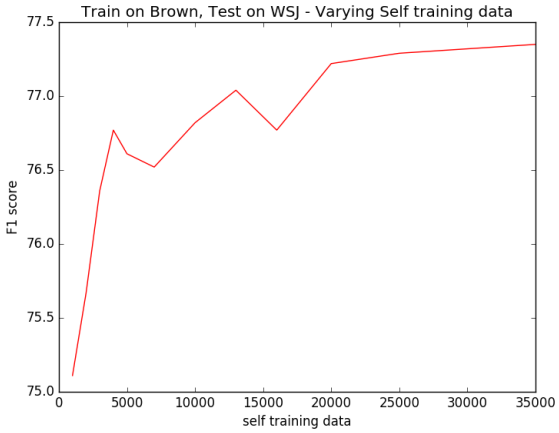


Varying Self Training Data

Now keeping the seed data fixed and varying the WSJ self training data we get the following plot.

1k	2k	3k	4k	5k	7k	10k
75.11	75.67	76.36	76.77	76.61	76.52	76.82

13k	16k	20k	25k	30k	35k
77.04	76.77	77.22	77.29	77.32	77.35



V. Discussion

The following observations were made that fit the following domains. I also state the justification for the observation below:

I. In-domain v/s Out-of-domain Testing

If we just consider the out-of-domain testing without self training and the in-domain testing, then for the case when WSJ was the seed, the in-domain performance on the average is 8-9% above the out-of-domain's performance in terms of F1 score. In the case of Brown seed, we see a similar trend, as the in-domain testing consistently beats the out-of-domain testing by about 6 - 8 %.

Reason:

This is expected, and quite nicely confirms that Syntactic Parsing on PCFG overfits to the domain of the data it is trained upon. Since WSJ follows a particular style of delivery that is common to the news, and Brown has a different style of delivery, more like articles and books, the parser trained on one model inherits that style. Each style has unique delivery, and thus varying probabilities for same Productions or totally different productions altogether. Since the end product of training are this Probabilistic Grammars, they inherently are tuned to the domain trained upon, and thus best parse new sentences from that domain.

They don't totally fail, as the delivery mode is still English, and thus even very different domains follow grammatical rules a lot of times. So this common Grammar helps us generalize to other domains, but never beats the present domain.

II. Unsupervized Domain Adaptation on Out-of-domain Testing

The unsupervized domain adaptation led to improvement in the performance on out-of-domain data. Though it is not enough. I hoped for a larger boost, as it takes a lot of out-of-domain context during self-training, but the performance actually rose on the average by mere 1-2% when the out-of-domain data was Brown, and rose by 4-5% when the out-of-domain data was WSJ when compared to the case without self-training. The difference in performance jump in both the cases can be justified, as I do below.

Reason:

The increase in the performance of the parser on the out-of-domain data, when part of the out-of-domain data is used for self-training, is because of the fact that now while re-training the parser, we are incorporating the style of the out-of-domain data, as now certain production which were a rare occurance in the in-domain data may now be more frequent in the out-of-domain data. This skews our probabilistic grammar to fit the test data, and thus the performance increases.

Now the reason for a greater increase in the performance when WSJ was out-of-domain is that, WSJ is more domain specific. All of its information is not captured by the more diverse brown data. Thus self-training makes the parser, more adaptive to the news domain and thus performance increases more. When

Brown is out-of-domain, due to the presence of too many domain this benefit is lost.

III. Effect of varying Seed Data

In general varying the seed data increased the performance in both the cases, when WSJ is seed and when Brown is seed. However the performance improvement tends to saturate as we increase the data. This trend is observed regardless of the scenario whether self-training occurs or not, or it is the in-domain testing or out-of-domain testing. The justification of the saturation is shown below.

Reason:
There is usually a small set of data that is representative of the entire dataset, be it style wise or grammar wise. Thus once the parser has obtained these information, it has learnt enough, and the new data does not help much. Thus the initial learning is more than the later learning. This is true for almost all Learning problems, and Natural Language Processing is no different.

IV. Effect of varying Self-Training Data

Varying self training data for the case when WSJ is the seed and Brown is the self-training data, leads to increase in the performance upto some extent, but then starts to fall. In the case when Brown is the seed data, and WSJ is the self-training data, the curve is not a nice inverse U curve as the case with Brown as self-training, but it tends to fall at times, and then almost saturates such that even a large increase in self-training data increases the performance by only 0.1-0.2%. The justification is as follows:

Reason:
Though self-training data helps in making the Probabilistic Grammar incorporate the production usage context of the out-of-domain data, but only a part of out-of-domain data is necessary for that. Beyond that the excess out-of-domain data just adds noise to the parser model. This happens because we dont have the correct parse for the self-training data. It is trained using the parser trained from the in-domain data, and thus this parser will give an inaccurate result for the out-of-domain parse. This inaccuracy grows as the self-training data is increased, and thus the accuracy of the parser suffers. This led to the saturation in case of Brown data as self-training data, and decrease in performance after some time, when the WSJ data was the self-training out-of-domain data.

V. Effect of Source-Target Inversion

There were many effects of source target inversion, some of whose reasons are stated above in the previous topics in Discussion (mentioned accordingly):

- Inverting the source and target leads to the performance increase for self-training data as compared to the testing data. (Explained in subsection II)
- The in-domain performance for data when WSJ is seed is higher in comparison to when Brown is seed.
  Reason:
  This is justified as the parser needs to recognize 8 different grammars, when tested on the Brown data, which is a more difficult task. However it has to just understand the news data when the in-domain testing is performed on the WSJ data.

VI. Comparison to the Reichart and Rappoport OI Benchmark

I observe the following relationship in comparison to the Rapport paper. They are as follows:

- The accuracy is more or less similar for both the WSJ case and Brown case.
- The case where the seed data is Brown and the self training data is WSJ, increasing the self-training data leads to a more abrupt wave like pattern in my experiment. This is similar to the ones observed in the original paper. However I have no justification to this. But now i am sure, it is not an issue with my implementation. The case when self-training is on Brown has a more smooth curve. However in the original paper the performance did not fall but saturate. I guess it is becasue the brown data might be chosen differntly for self-training in the original paper.