Neural Dependency Parsing with "Unsupervised" Domain Adaptation

Nayan Singhal (ns28844)
University of Texas at Austin
Austin, TX 78712
nayans@utexas.edu

Abstract

Creating a large amount of annotated data for training the parser is very expensive and the performance decreases if we use training and testing data from different domains. Since learning statistical parser can be quite specific to the genre of the training corpus. In this assignment, we use self training in order to adapt the system trained on one domain to perform better on the other domain, using only a small amount of the annotated data. In this task, we train our model on Wall Street Journal corpora and self-train on brown corpora and achieved the overall performance gain of 1.7% on brown testing dataset.

1. Introduction

Most of the previous parsers were trained on the annotated data which is very expensive to create and takes a lot of time. The performance of the parser decrease if the distance between the genres of the training and testing data increases. Hence, training the parser on the very small annotated data is very important when both seed and test data are from the same domain (In domain) or seed and test data are from different domain (Out domain).

Self training is the method which can train the parser using unannotated data by using small seed data. The model is first trained on the small annotated seed data which is very easy to made (just consisting of some 1000-2000 sentences), and then the model is used to annotate the self-training data (unannotated data) and then the manually and automatically annotated data is merged to fine-tune the same model. Since we used the small annotated data and unsupervised data is easily available, the self training method is highly used as compared to training the model only on supervised data.

In this assignment, we use self-training to enhance the performance of the PCFG parser that is trained on small annotated dataset, wsj (brown) and self train it with the brown (wsj) dataset and test the model for in-domain and out-domain. We perform two experiments, examining the effect on the performance of the model both in domain and out-domain:

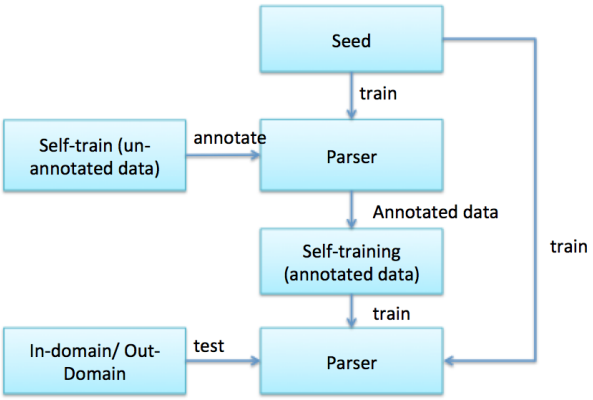

Figure 1: Unsupervised Domain Adaptation model: seed data is used to train the model which is used to automatically annotate the self-training dataset. The seed dataset is merged with the automatically annotated data which is further used to re-train the model.

1. Varying the seed data size and maintaining the self-training size constant.
2. Keeping the seed size constant and varying the self-training data.

The main contribution of this work is that it shows how to train the model when you have sufficient manually annotated data in one domain but have very less or no annotated data in other domain. Our results shows that self-training is a valuable technique for improving the performance of the statistical parser when the seed data is very less.

2. Methodology

The methodology which is shown in fig 1 is divided into two step process:

2.1. Dataset Preprocessing

We have two dataset for training the model: wsj (Wall Street Journal text) and brown (Brown corpus of mixed-genre text) corpora. First, we split the wsj data from 02-10 into 90-10 train and test ratio and similarly, for brown corpus, we split the train and test 90-10 for each genre. For splitting the dataset, we calculate the total sentences in the corpora and divide accordingly.

2.2. Train the model

We trained our parser on the seed data (wsj dataset) that consists of very less annotated data. The model trained with the small dataset are used to parse the self-training dataset (brown corpus), that contains several thousand sentences. The automatically annotated data is merged with the manually annotated seed training data which is then used to retrain the parser. The retrained parser are then used for testing on in-domain and out-domain corpus.

3. Experiments

We perform four experiments by training the model for 500 iterations which are explained in the following subsections:

3.1. Seed: wsj (varied), Self-Training: brown (fixed)

We train the model using seed: wsj dataset by varying it from 1,000-14,000 sentences and fixed the brown 90% as the self-training dataset.

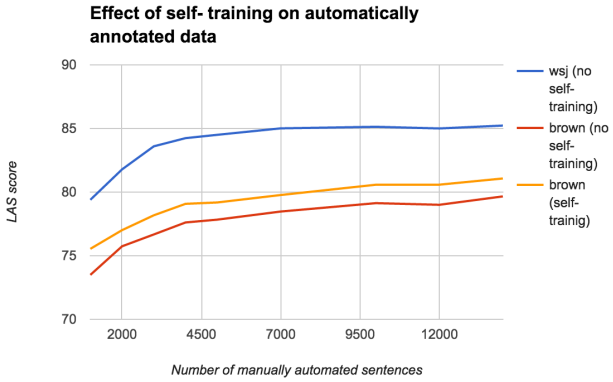

Figure 2: Model 1: Varying the seed(wsj) dataset from 1,000 - 14,000 sentences.

From the fig 2, we can see that there is significant performance drop from in-domain (blue curve) to out-domain (red curve) testing, almost (5.56%). As, we can see from the graph, initially when the seed training is very less, the LAS score between the in-domain and out-domain is almost (5.90%). But, when we increased the seed data, it decreases to (5.56%). And, its expected because when you increases the seed data, it makes the model more adaptable to wsj and will help in better performance on in-domain testing, but slightly less performance on out-of domain testing.

3.2. Seed: wsj (fixed), Self-Training: brown (varied)

We train the model using seed: fixed the wsj dataset to 10,000 sentences and varied the brown as the self-training from 1,000 - 21,000 sentences.

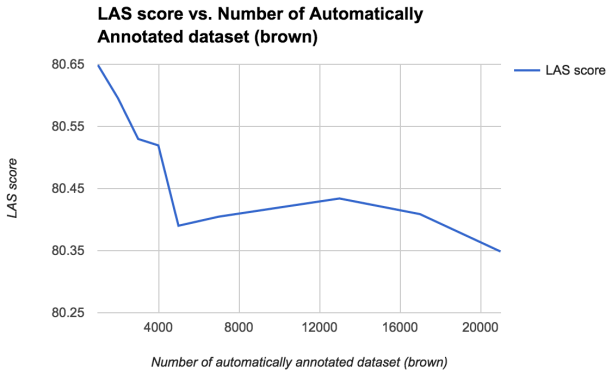

Figure 3: Model 1: Varying the self-training(brown) dataset from 1,000 - 21,000 sentences.

From the fig 3, we can see by fixing the seed size and varying the self-training size, the performance of the model decreases from 80.65% to 80.35%. And, its expected because with more unannotated sentences, the probability of the correct label sentences will decrease which makes the model train relatively worst as compared to the model trained with fewer self-training sentences.

3.3. Seed: brown (fixed), Self-Training: wsj (varying)

We train the model using seed: brown dataset by varying it from 1,000 - 21,000 sentences and fixed the self-training wsj dataset as 90% of training data.

From the fig 4, we can see that there is significant performance drop from in-domain (blue curve) to out-domain (red curve) testing, almost (4.3%). Also, we can see from the graph, initially when the seed training is very less, the LAS score between the in-domain and out-domain is almost (2.5%). But, when we increased the seed data, it decreases to (4.3%). And, its expected because when you increases the seed data, it makes the model more generic (not too much) and will help in better performance on out-domain testing.

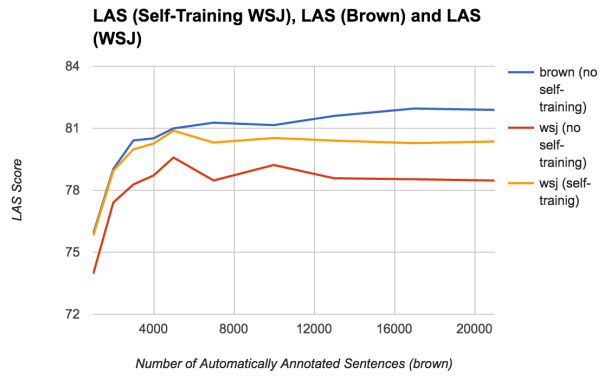

Figure 4: Model 1: Varying the seed(brown) dataset from 1,000 - 21,000 sentences.

3.4. Seed: brown (fixed), Self-Training: wsj (varying)

We train the model using seed: fixed the brown dataset 90% and vary the wsj data as self-training data from 1,000 - 14,000 sentences.

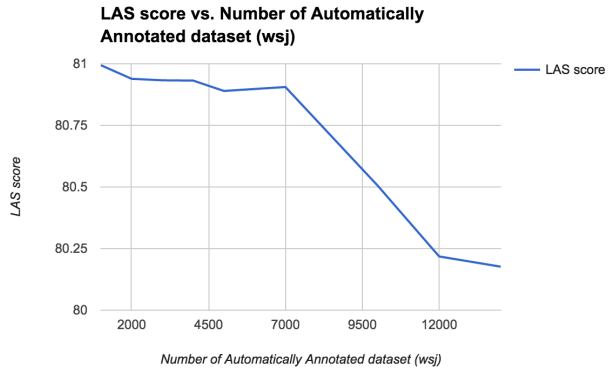

Figure 5: Model 1: Varying the self-training(wsj) dataset from 1,000 - 14,000 sentences.

From the fig 5, we can see by fixing the seed size and varying the self-training size, the performance of the model decreases from 80.99% to 80.17%. And, its expected because with more unannotated sentences, the probability of the correct label sentences will decrease which makes the model train relatively worst as compared to the model trained with fewer self-training sentences. And, its expected because with more unannotated sentences, the probability of the correct label sentences will decrease which makes the model train relatively worst as compared to the model trained with fewer self-training sentences.

4. Discussion

There are some findings which we want to describe in the following subsection:

4.1. performance drop from in-domain testing to out-of-domain testing

From the fig 2, we can see that there is significant performance drop from in-domain (blue curve) to out-domain (red curve) testing, almost (5.56%). As, we can see from the graph, initially when the seed training is very less, the LAS score between the in-domain and out-domain is almost (5.90%). But, when we increased the seed data, it decreases to (5.56%). And, its expected because when you increases the seed data, it makes the model more adaptable to wsj and will help in better performance on in-domain testing, but slightly less performance on out-of domain testing.

4.2. Impact of unsupervised domain adaptation on the performance of out-of-domain testing

As we can see from the fig 2, the model trained with the unsupervised domain adaption (blue curve) performs better as compared to the model trained without unsupervised domain (yellow curve). Gain is less (4.13% - 71.5% to 68.66%) initially when we are using 1000 sentences but it increases till 4.62% (77.70% - 74.25%) in the end when we are using 14,000 manually annotated data. And, its expected because it makes our model more accurate since we are using large manually annotated sentences.

4.3. Impact of the seed size and self-supervised training sets on the performance

As we can see from the fig 2, with increase in the seed size, the performance of all the three graph increases (in domain testing, out-of domain and unsupervised domain adaptation in-domain testing). And its expected, with increase in wsj seed size, model should predict the correct parses for the wsj sentences (in-domain testing). It should increases the performance of the brown sentences (both out-domain and unsupervised domain adaptation testing) as explained in the previous answer. There is a boost in the performance of the model initially but it gets taper off when we vary the seed size from 10,000 - 14,000. It shows that even if we increase the seed size, we wont get much boost in the performance of the model.

From fig 3, we can see by fixing the seed size and varying the self-training size, the performance of the model decreases from 80.65% to 80.35%. And, its expected because with more unannotated sentences, the probability of the correct label sentences will decrease which makes the model train relatively worst as compared to the model trained with fewer self-training sentences.

4.4. Impact on the performance by inverting the source and target dataset

As, we can see from the fig 2 and fig 4, the overall figure shapes are almost same. With increase in seed data, the performance of all the three graph increases (in domain testing, out-of domain and unsupervised domain adaptation in-domain testing). The important point to notice at, when we take 10,000 sentences for wsj (brown) as seed set and test it on wsj and brown, we get the performance of 85.13% (79.22%) and 79.13% (81.15%) respectively. There is a big gap between the performance of wsj and brown testing when wsj and brown seed sentences are used.

Similarly, when we do unsupervised domain adaptation while taking wsj and brown as seed data, the performance of the model on brown(80.38%) and wsj (80.52%). This shows that out-of-domain testing is better when we used brown as the seed data. And it makes sense, because brown contains multiple genre and is more diverse as compared to wsj which is more restricted. Hence, its easier to adapt to wsj as compared to brown.

4.5. Result Comparison with Reichart and Rappoport paper for the OI setting

Plots obtained for the Stanford parser in our experiment are very similar to the plots given in the paper for the Collins parser. With increase in the manually annotated data, the performance of all the three graph increases (in domain testing, out-of domain and unsupervised domain adaptation in-domain testing). Also, we can see that the model trained with the unsupervised domain adaption performs better as compared to the model trained without unsupervised domain.

Since they were using F-score and we are using LAS score, so we cant compare the results quantitatively.

5. Conclusion

Our results reinforce the claim made in the paper that self-training is a valuable technique for improving the performance of the statistical parser when the seed data is very less. These days we have seen a lot of researcher in a generalization of the technique where they trained the model on one dataset and fine-tune the model for the other task where they have less training data.

To future work, we would like to check the train the model by combining the self-training data from different domains.