Statistical Parsing with Unsupervised Domain Adaptation

Kunal Lad : KL28697
kunal.lad@utexas.edu

Overview

Creating large amounts of annotated data to train statistical PCFG parsers is expensive, and the performance of such parsers declines when training and test data are taken from different domains. The task of *domain adaptation* or *transfer learning* is to adapt a system trained on one "source" domain to perform better on a new "target" domain. Typically, one assumes that there is sufficient labeled training data in the source, but there is little or no labeled data for the target, since gathering sufficient training data in every new domain is expensive and labor intensive. In this assignment, we will look at the special case of "unsupervised" domain adaptation, where one must adapt a parser trained on sufficient labeled data in the source (e.g. WSJ) to a new domain (e.g. Brown) where only *unlabeled* training data is available.

Experiments

Datasets: For Brown corpus we create a 90-10 split for generating training/test data. For WSJ corpus we use sections 02-22 for training and section 23 for testing. If we need to select a subset of k sentences from training data, then we select k sentences randomly if selection has to be made from brown training corpus or select first k sentences from WSJ training corpus.

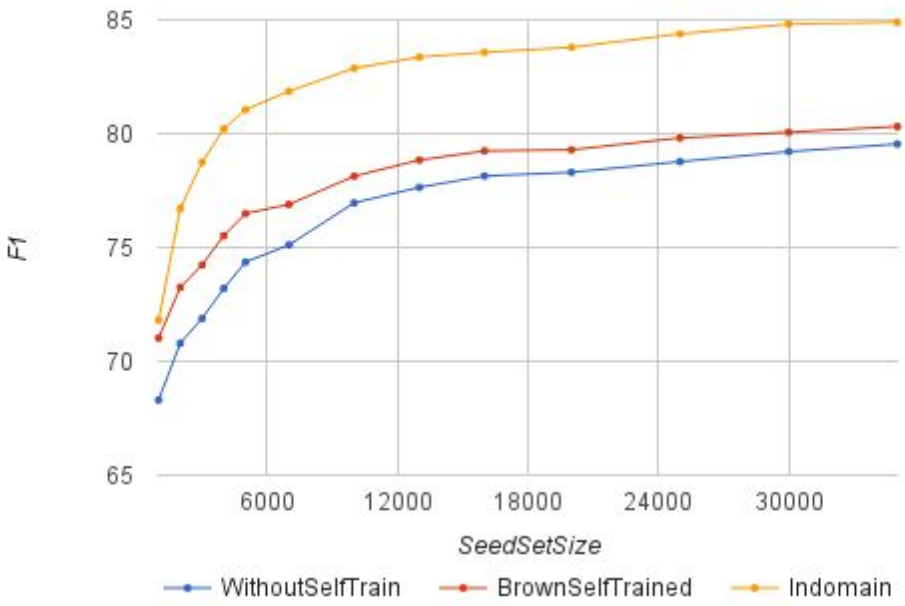

Fig 1: Performance comparison of PCFG trained on WSJ corpus with and without unsupervised domain adaptation for Brown Test Set against indomain WSJ Test Set.

We used Stanford Parser implementation for performing all the experiments in this assignment. We performed the following experiments for investigating the following:

• How much does performance drop from in-domain testing to out-of-domain testing?

Plots in Fig1 shows that there is a significant performance drop (~5%) from in-domain testing to out-of-domain testing. As expected the difference between in-domain and out-of-domain performance is very small when seed set used for training is small. It keeps increasing continuously upto a point (around 7000) and becomes almost a constant after that. This clearly shows that model does not perform as well on out-of-domain data.

• How does unsupervised domain adaptation impact performance on out-of-domain testing?

We can see from plot in Fig 1 that model with unsupervised domain adaptation (red curve) performs better than model without unsupervised domain adaptation (blue curve). Gain in performance is high when seed set is small (71.02% vs 68.3% for seed set of size 1000) but it starts reducing as the size of seed set increases. For larger values of seed set the gain is small (80.31% vs 79.54% for seed set of size 35000). This variation in gain with seed set size is understandable because for model trained with small seed set, amount of unsupervised domain adaptation data is significant in size when compared to model with large seed set.

• How does increasing the size of seed and self-supervised training sets affect the relative performance?

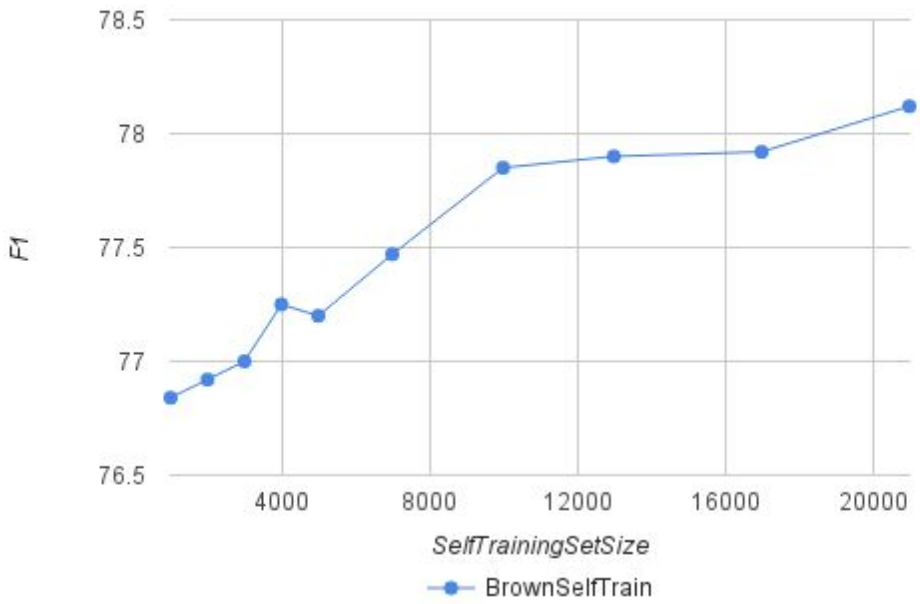

Fig 2 : Performance variation with in increase in size of self training data for unsupervised domain adaptation for brown corpus

From plot in Fig 1 we can see that performance improves for all 3 models (with and without unsupervised adaptation for out-of-domain testing and in-domain testing) with increase in seed set size. There is a huge boost in the performance in the beginning but the curve starts to taper off for all 3 models around 25000 after which point adding more training data doesn't give any significant benefit. Plot in Fig 2 shows that performance increases with increase in self-supervised training set. Although both plots show improvement in performance with increase in size of seed/self-training data, plot in fig 1 rises sharply in the beginning and tapers off after that. On the other hand in fig 2 we see performance

improving almost linearly till ~10000 before slowing down. However the curve from 17000 - 21000 range suggests that it has not converged and there is still scope for further improvement.

• How does inverting the "source" and "target" impact your results and why?

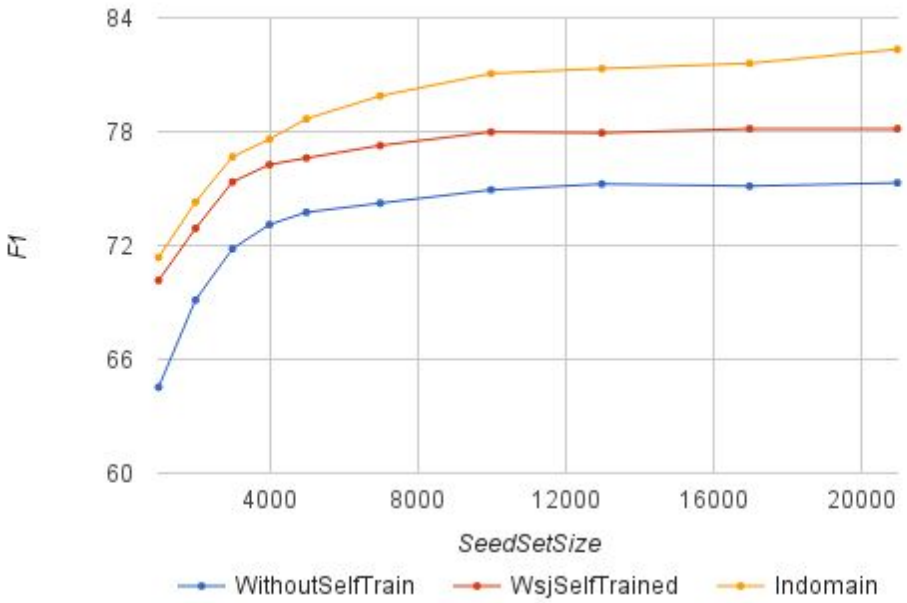

Fig 3 : Performance comparison of PCFG trained on Brown corpus with and without unsupervised domain adaptation for WSJ Test Set against indomain Brown test set.

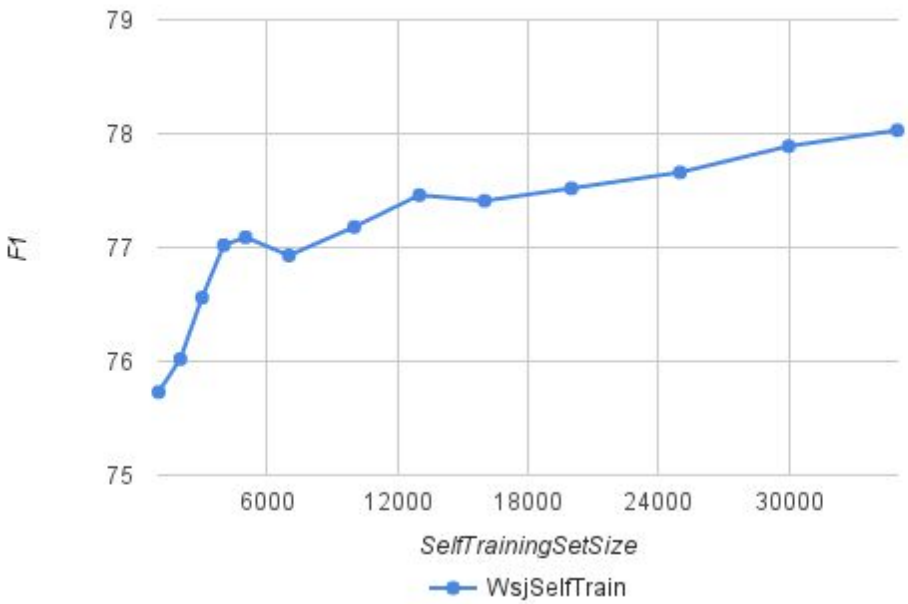

Fig 4 : Performance variation with in increase in size of self training data for unsupervised domain adaptation for brown corpus

By comparing plots in Fig 1 and Fig 3 we can see that overall curve shapes are similar after reversing source and target corpus. However by comparing model performances with (red curve) and without (blue curve) unsupervised domain adaptation in Fig 1 and Fig 3 we observe that unsupervised domain adaptation helps more when Brown is used as source corpus and WSJ as target corpus. Reason for this is that Brown corpus has multiple genres and is more diverse than WSJ which is restricted to a particular domain. Hence it is easier to adapt to WSJ as compared to Brown.

From plots in Fig 2 and Fig 4 we can see that performance variation with respect to self training data size also has similar curve as before.

• How do your results compare to the results described in the Reichart and Rappoport paper for the OI setting?

Plots obtained for Standford Parser in our experiments are very similar in shape to those given in paper for Collins parser for OI setting with the difference that range of seed set for Collins parser is 10 to 2000 sentences but for our experiments the range of seed set varies from 1000 to 35000. By comparing the plot for Collins parser in paper (Fig 3) with our plots (Fig 1 and 3) we can observe that performance gain due to unsupervised domain adaptation is around 3-4 % for both the parsers when seed set size is 2000 (Largest value of seed set for Collins Parser).

NOTE: Detailed results of all experiments can be found here

Conclusion

Our results reinforce the claim made in the paper that self-training is a valuable technique for improving performance of generative statistical parsers, especially when seed data is less. Recently we have started seeing a lot of success in a generalization of this technique where a base model trained for some application is later on fine-tuned with small amounts of labelled data for another application. This is particularly prominent in NLP and Computer Vision fields where Deep Learning Model trained on large amount of data for some application is fine-tuned for other applications with small amount of data later on.