Homework 3 - Statistical Parsing with Unsupervised Domain Adaptation

Abhishek Sinha

I. Problem Statement

The objective is to improve the accuracy of statistical parsing on a new domain through unsupervised domain adaptation.

II. Approach

The approach followed was that of *self training* as described in the ACL paper by Reichart and Rappoport . In this approach, a parser is first trained on the source domain for which there exists labelled data. The parser thus generated is used to predict labels (hard) for the unlabelled self-training data. Then the original out-of-domain labelled data is combined with the self-training data (with predicted labels) to generate a new dataset of pseudo-labelled data. Finally, a parser is retrained on this pseudo-labelled data and predictions are made on the test target domain data.

III. Implementation

A common *JAVA* interface named - *LexicalizedParserCommandLineInterface.java*was developed which was used for performing the experiments in all 3 parts (part 4, 5 and 6). This interface takes the following six parameters as input

- *Source Seed Data Path*
- *Number of Source Seed Sentences to use*
- *Target Self Training Data Path*
- *Number of Self-Training Sentences to use*
- *Target Test Data Path*
- *Number of Test Data Sentences to Use*

Note - In the above interface, any of the number of sentences parameters can be set to 0 which allows us for example to not use any training data at all. Additionally, we can use all the sentences by giving -1 as input. Given this interface, the approach can be described as follows:

- The source, self-training and test treebanks (Memory Treebanks) are loaded using the provided paths.
- Each of the 3 tree banks is filtered using the specified number of sentences
- The Lexicalized Parser is first trained on the source memory tree bank.
- Predictions are made on the test tree bank using this parser to handle the in-domain or the no-self-training data case.
- Then predictions are made on the sentences extracted from the self-training memory treebank to generate a new treebank.
- The above treebank is then merged with source memory tree bank.
- Finally the Lexicalized Parser is again trained on this merged tree bank and predictions are again made on test memory treebank.

In addition to the above basic JAVA file, the following additional scripts were implemented to pre-process data, generate results and post-process the results

- *preprocess_data* This is a python script which preprocess the Brown Dataset to produce a single big training file and a single testing file. In additional to this 90% of the sentences are selected from each genre and added to a list of training sentences while the remaining sentences are added to a list of test sentences. The list of training sentences thus generated is then randomly shuffled. This ensures that for any given total size, we will approximately choose sentences from the different genres in the same proportion as their total sizes.
- *Condor Scripts* These scripts were used to submit the various *JAVA* jobs to the Condor cluster.
- *process_results* This is a python script which reads the log files produced, extracts the f1 values from them, aggregates the f1 values across different sizes and then finally produces plots of the learning curves.

IV. Experiments

We conducted the following 8 experiments in total.

- Take WSJ as the source as well as target data. Train on sections 02-22 varying the seed data size from 1000 to 35000 and test on section 23. Plot F1 score
- Take WSJ as source data and Brown as target data. Train on sections 02-22 varying the seed data size from 1000 to 35000 with no additional self-training. Test on Brown test data (10% from each genre) and plot F1 score.
- Take WSJ as source data and Brown as target data. Train on sections 02-22 varying the seed data size from 1000 to 35000 and do self-training using all the unlabelled training data from Brown (90% from each genre). Plot F1 score.
- Take WSJ as source data and Brown as target data. Keep seed data size as 10000 sentences. Vary the self-training data size (Brown) from 1000 to 21000 and test on Brown. Plot F1 score.
- Take Brown as the source as well as target data. Vary the seed data size from 1000 to 21000 and plot F1 score
- Take Brown as source data and WSJ as target data. Vary the seed data size from 1000 to 21000 with no self-training and plot F1 score.
- Take Brown as source data and WSJ as target data. Vary the seed data size from 1000 to 21000 and do

self-training using all the training data from WSJ (all sections from 02-22). Plot F1 score.

- Take Brown as source data and WSJ as target data. Keep seed data size as 10000 sentences. Vary the self-training data size (WSJ) from 1000 to 35000 and plot F1 score.

V. Results

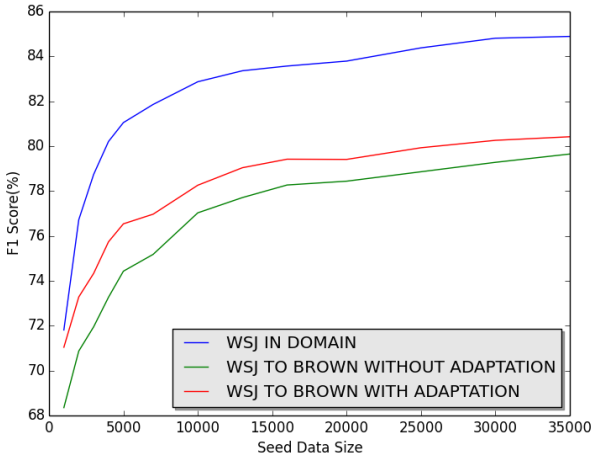

Fig. 1: Learning Curves with respect to seed set size variation for source = WSJ, target = Brown

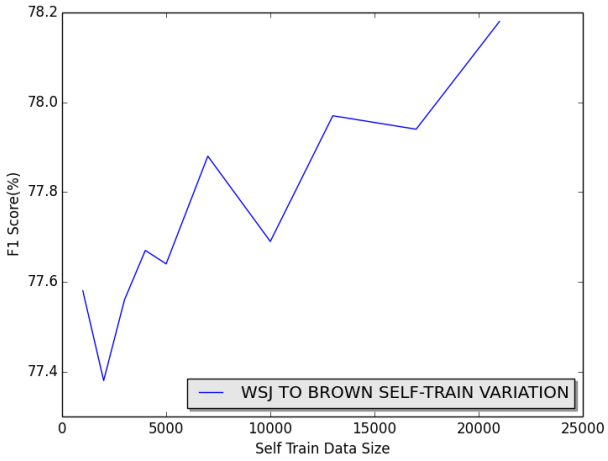

Fig. 2: Learning Curves with respect to self training set size variation for source = WSJ, target = Brown

VI. Discussion

- **How much does performance drop from in-domain testing to out-of-domain testing?** When WSJ is used as the source domain and Brown as the target domain, the performance drop varies from 3.5 to 7% depending on the size of the seed dataset. When Brown is used as the source domain and WSJ as the target domain, the performance drop varies from 1.8 to around 9.8%.
  In the case of WSJ as source domain and Brown as target domain, initially drop in performance is less (3.5%), increases to the maximum value of around 7% at seed size of 4000 and then decreases. In the case of Brown as source domain and WSJ as the target domain the maximum drop in performance occurs at 1000 points and then decreases as the number of points is increased. The drop in performance on moving to a new domain is because the source domain cannot model some of the regularities present in the target domain and hence a parser trained on the source domain will have a reduced performance when tested on the target domain in comparison to testing on the source domain itself.

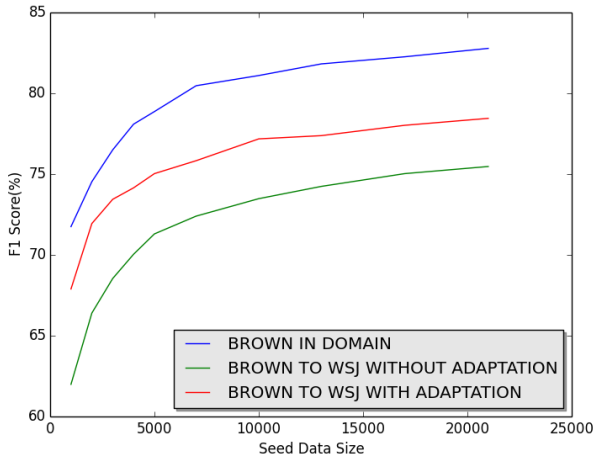

Fig. 3: Learning Curves with respect to seed set size variation for source = Brown, target = WSJ

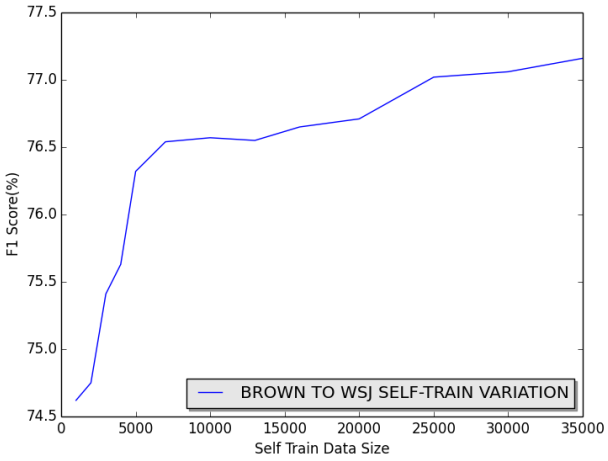

Fig. 4: Learning Curves with respect to self training set size variation for source = Brown, target = WSJ

- **How does unsupervised domain adaptation impact performance on out-of-domain testing?** Unsupervised domain adaption helps in improving the performance on out-of-domain testing. For WSJ as source domain and Brown as target domain, we are able to reduce the drop in performance to around 0.8% for seed size of 1000. Even the worst drop in performance is now reduced to around 5% from the earlier figure of around 7%.

Similar gains in performance are also seen for Brown as source domain and WSJ as the target domain. The worst case drop in performance is now around 4.6% from the earlier figure of around 10%.

Another thing we observe is that the improvement of performance is generally larger when the size of the seed data set is smaller. For example, when WSJ is source domain and Brown is target domain, we see improvements of around 2.5% for the small seed set sizes but the improvement for larger seed set sizes is less than 1%. Similarly for Brown as source domain and WSJ as the target domain, we see improvements of almost 6% whereas for larger seed set sizes the improvement is closer to 3%. This also makes intuitive sense since when we increase the size of the seed data set, it will be able to model the regularities of the target test domain to an increasing extent. So the marginal gain from using the self-training set will be lesser because there are lesser additional regularities it can model since most of them would already be modeled by the source domain data.

- **How does increasing the size of seed and self-supervised training sets effect the relative performance?** Both increasing the size of the seed dataset and the size of the self-training dataset leads to improvement in the overall performance. For both WSJ as seed and Brown as seed, there is an improvement in performance of around 12% as we increase the size of the seed data set from the minimum to maximum.

  In case of self-training dataset size, when WSJ is source domain and Brown is target domain, increasing the size of the self-training dataset leads to an improvement of around 0.6 to 0.8%. On the other hand, for Brown as the source domain and WSJ as the target domain we se an improvement of around 2.5-3 %. This is because Brown dataset contains much more diverse data in comparison to the WSJ dataset. It is tougher for unsupervised domain adaptation to learn from this more diverse data in comparison to learning from the restricted financial news data present in WSJ.
- **How does inverting the "source" and "target" impact your results and why?** When Brown is used as the source domain and WSJ as the target domain, the improvement in performance through unsupervised domain adaptation is much higher in comparison to when WSJ is the source domain and Brown is target domain. This is because WSJ has more restricted financial news related data where Brown has much more general, diverse data. Thus it is easier to learn in an unsupervised way when there is a lot of data the same kind than when there is a lot of diverse data.
- **How do your results compare to the results described in the Reichart and Rappoport paper for the OI setting?** Our results match the results in the Reichart and Rappoport paper for the OI setting. The learning curve we learn by plotting F1 score against the number of seed data sentences has a similar shape to the curve obtained by the authors in the Figure 3 of their paper where they study the OI setting. We too find that for as seed data size is increase, the performance initially improves rapidly and then saturates. In addition, we too find that the improvement in performance by virtue of self-training is much higher when using small seed data size than when using a large seed data size. We cannot compare with the table 4 of the paper since there the number of seed data sentences is mostly less than 1000 while we take a minimum of 1000 seed data sentences.