Team Mitron - Mid-Term Project Report [*]

Ajinkya
130101004
IIT Guwahati

Jainam Shah
130101025
IIT Guwahati

Prateek Raisinghani
130101059
IIT Guwahati

Kushal Chawla
130101082
IIT Guwahati

1. ABSTRACT

Preposition sense disambiguation is at the heart of complex NLP tasks like machine translation. They are full of ambiguity and their sense is critical to understanding the meaning of a sentence. The problem at hand is to identify the correct sense of a given preposition and its context. We aim to use a transfer learning approach by combining general word sense disambiguation with the task of preposition disambiguation. We have implemented several frequency based baselines, off the shelf classifiers, feed forward and recurrent neural networks. In this report, we describe the problem, our approach and summarize our results obtained till now.

2. INTRODUCTION

Here we give a brief introduction to the issues and difficulties with the task of preposition disambiguation as well as some of the classifiers and approaches that have been worked with before for the task.

Preposition disambiguation is considered to be the most difficult class of problem in NLP. The main task to be achieved is the word sense disambiguation of prepositions. Prepositions can have different meanings based on the contexts they are a part of. As an example consider the statement :- "Kushal met Prateek **at** the room." and "Kushal met Prateek at 10 p.m.". In both the sentences, the preposition **at** is used but in the first case it refers to the spacial locality where as in the second case it refers to temporal locality.

This was the complexity in the problem. Another kind of issue is the one related to the availability of a dataset. There is a scarcity of labeled dataset specifically for the case of prepositional disambiguation and so it has given rise to semi supervised approaches and so is our approach of transfer learning. Moreover, the datasets have been limited to very new number of common prepositions. Newer datasets are coming up, like the web reviews corpus we use for this work, which now include several less common prepositions and pseudo prepositions(multi word) as well.

Knowing about the issues and difficulties of the problem, we now try to see the methods that have been tried and tested for the purpose of preposition disambiguation.

The first set of models proposed used a supervised approach. The first supervised approach used different kinds of syntactical and semantical features for the purpose of representation and finally used a maximum entropy based model for classification[7]. The next model took the following approach. Given a context, they used a language model to predict which substitute is most likely[8]. And the meaning of the preposition would be based on this likeliness value.

In most of the approaches the general trend has been as follows. The main focus has been in trying to extract different features from the given corpus. Once the feature representation has been deduced then the methods of regression, support vector machines, neural nets and that of decision trees can be applied. Thus various pre-processing tools such as WordNet has been used to come up with information pertaining to the text for feature representation.

Going into a little detail of the model used by one of the papers, a LSTM encoder is used in order to predict what the translation of a preposition would be and then later fine tune it to show that using data that has not been annotated can be used improve the task of disambiguation of preposition[1]. Another approach used quite often is that of the knowledge approach in which we also require the outputs from tools such as StemCor, Wikipedia etc but because the task of preprocessing the corpus through such tools is tie consuming and resource intensive, they are not that widely used [5].

3. OUR PROPOSED METHOD: TRANSFER LEARNING APPROACH

Supervised corpora available for preposition disambiguation are very small, owing to the fact that they are expensive to prepare. Previous approaches as discussed above, have used different knowledge bases to compensate for lack of available data.

We suggest a transfer learning approach[3] for this task. We aim to combine a general word disambiguation task with preposition disambiguation where some parameters will be shared between the two tasks. A prototype of the model is given in the figure 1.

Given a sentence and a preposition, on the input side, we consider the left side context, preposition and right side contexts separately. The contexts are passed through recurrent layers, be it standard RNN, LSTM or GRU. Then, we perform a pooling operation, either max pooling or average pooling. Then we concatenate the outputs together along with the representation of the preposition(can be multi-word). The parameters in this layer are shared between both the word disambiguation and preposition disambiguation tasks. In the next layer, we have separate full connections for each of the tasks, followed by respective softmax layers.

We plan to pool the datasets of both the tasks together and train the model from pooled samples. The intuition behind such an architecture is that the recurrent model tries to represent the context, whether for a word disambiguation or a preposition. Hence, we expect the model to learn similar parameters in this layer for both the tasks.

While training, if a sample from word disambiguation appears on the input side, it will update the corresponding fully connected layer weights and the shared parameters in the recurrent layer. Similarly, for a sample in preposition disambiguation, it will only train the corresponding fully connected parameters and recurrent layer parameters. This solves the problem of data scarcity where the parameters in the recurrent layer will be trained using the samples from both the datasets.

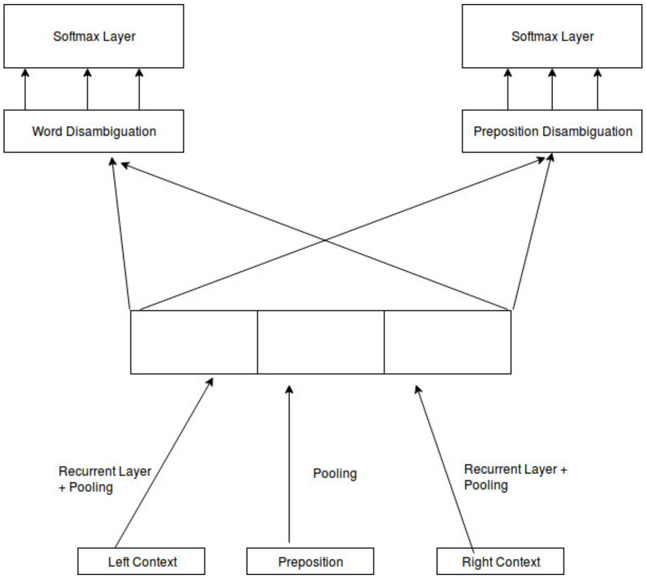

Figure 1: Model architecture for transfer learning approach

4. EXPERIMENT

In this section, we discuss the word embedding and datasets used in the experiments. Specific details of the model have been given mentioned along with their results in the next section. For further details, we can share the code when required.

4.1 Word Embedding

For this work, we have used pretrained GloVe([4]) 50 dimensional word embedding. The vectors were trained on Wikipedia corpus with 6 Biliion tokens, with a vocabulary of 400 thousand words. We have processed the dataset to find as many as pre-trained word embedding as possible. Out of the vocabulary of 5166 of the Web Reviews corpus, 4961 words have their pretrained word embedding available. The words for which no embedding was found, were initialized randomly.

4.2 Datasets

We use two datasets for this work. As with [1], our main focus would be on the Web Reviews corpus[6] but we will also do the analysis on SemEval 2007 corpus[2] to compare our model with previous approaches:

4.2.1 SemEval 2007 Corpus

This corpus contains 34 prepositions present across 16,557 training and 8096 testing samples. Each sample consists of a sentence and a marked preposition. Each preposition has its own set of senses of size 2 to 25.

4.2.2 Web-reviews Corpus

Our main focus is on the Web reviews corpus which uses a unified sense inventory for all the prepositions. It has 4250 samples with 114 distinct prepositions. There are 63 supersenses shared between all the prepositions. Figure 2 shows a frequency distribution of these supersenses. It can be observed that the distribution is highly skewed with a minimum and maximum frequencies of 1 and 596 respectively.

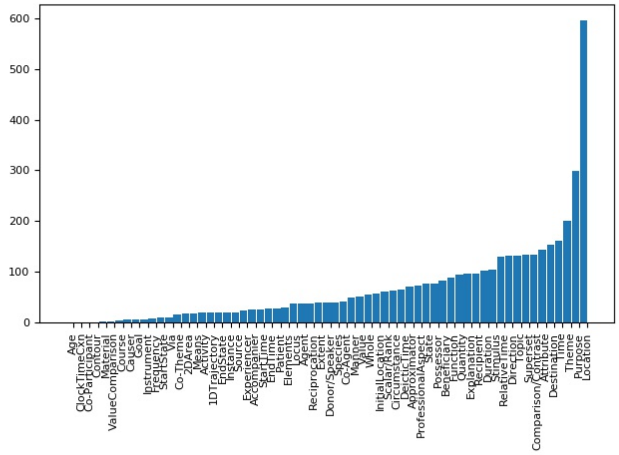

Figure 2: Frequency distribution of supersenses in Web-reviews Corpus.

5. RESULTS AND DISCUSSION

In this section, we give the performance of various models on Web Reviews corpus. These include a few baselines, simple classifiers, feed-forward and recurrent neural network models. We plan to implement transfer learning approach later.

5.1 Baselines

The models implemented were:

1. **MaxClass**: Find the supersense with maximum frequency in the training dataset. For any new sample, return this maximum frequency supersense as the predicted sense.

2. **MaxClassPrep**: Find the supersense with maximum frequency for each preposition separately in the training dataset. For a new sample, check its preposition and assign the corresponding maximum frequency supersense.

3. **RandClass**: Find the frequency distribution of supersenses over the training dataset. Convert this distribution to a probability distribution and assign the supersense to a new sample by sampling from this distribution.

4. **RandClassPrep**: Do the same as above but find the probability distribution for each preposition separately.

5. **UniformRandClass**: Similar to **RandClass** but use a uniform probability distribution instead of a weighted one.

6. **UniformRandClassPrep**: Similar to **RandClassPrep** but use a uniform probability distribution instead of a weighted one.

Model Name	Dev. Set	Test Set
MaxClass	0.135	0.136
MaxClassPrep	**0.411**	**0.425**
RandClass	0.040	0.044
RandClassPrep	0.275	0.277
UniformRandClass	0.020	0.024
UniformRandClassPrep	0.135	0.176

Table 1: Performance of various baseline models. The table shows the accuracy values(ranging from 0 to 1).

The performance of these models is summarized in table 1.

5.2 Off the shelf classifiers

For this task, we must process the data first and get input and output representations. We used pretrained GloVe 50 dimensional word vectors to represent the input. Given a sentence and a preposition in it, we divide the sentence in 2 parts- left side context and right side context corresponding to the position of the preposition in the sentence. We convert all the words to their corresponding word vectors. We then performed a pooling operation on the two contexts and concatenated the result with the pooled(in case of multi word) representation of preposition. This was done in the following two ways:

1. **Average Pooling** - In this case, we take the average value across all the vectors for each dimension.

2. **Maximum Pooling** - In this case, we take the maximum value across all the vectors for each dimension.

Model Name	Pooling	Dev. Set	Test Set
SVC	Average	**0.542**	**0.537**
KNeighborsClassifier	Average	0.424	0.425
RandomForestClassifier	Average	0.500	0.521
MLPClassifier	Average	0.535	0.492
SVC	Max	0.446	0.472
KNeighborsClassifier	Max	0.395	0.416
RandomForestClassifier	Max	0.511	0.525
MLPClassifier	Max	0.440	0.481

Table 2: Performance of various library models. The table shows the accuracy values(ranging from 0 to 1).

The four models from scikit learn library which performed better than others were SVC, KNeighborsClassifier, RandomForestClassifier, MLPClassifier. The performance of these models is summarized in table 2

Model Name	Pooling	Dev. Set	Test Set
Feed Forward NN	Average	0.495	0.508
Feed Forward NN	Max	**0.526**	**0.510**

Table 3: Performance of feed forward neural network models. The table shows the accuracy values(ranging from 0 to 1).

5.3 Feed forward neural network models

m The input and output representations were obtained in the same manner as in previous section. We implemented a single hidden layer, feed forward neural network in tensorflow for this task. The advantage here is that along with the weight parameters, we could also train the word vectors in the training process. The pooling again can be done in two ways which gives rise to two different settings. The hyperparameters used for this task were as follows:

1. Learning rate: 0.01

2. Embedding size: 50

3. Number of epochs: 100

4. Batch size: 100

5. Drop Out: 0.7

6. L2 regularization: 0.0001

7. Pooling: Avg/Max

The results are summarized in table 3.

5.4 Recurrent neural network models

For NLP tasks, where we are trying to capture the context of a preposition, a better choice than simple Feed Forward neural network models is to use Recurrent Networks. So, in this case instead of using a pooling operation on the input directly, we pass the left and right contexts via 2 different recurrent layers, and then concatenate the output with the representation of the preposition itself. The pooling layer after the recurrent layer is kept as Max Pooling. After the recurrent layer, we have a fully connected feed forward layer as usual.

Model Name	Dev. Set	Test Set
RNN	0.546	0.489
LSTM	0.557	0.523
GRU	**0.584**	**0.528**

Table 4: Performance of various recurrent models. The table shows the accuracy values(ranging from 0 to 1).

The hyperparameters used were:

1. Learning rate: 0.01

2. Embedding size: 50

3. Number of epochs: 50

4. Batch size: 100

5. Drop Out: 0.8
6. L2 regularization: 0.0001
7. Pooling: Max

We implemented these models in tensorflow. Three models were used, namely, RNN, LSTM and GRU. The results are summarized in table 4

The results obtained till now are not satisfactory though some more effort may be made in hyperparameter tuning. This may be attributed to lack of sufficient data and the skewed frequency distribution as given before. Instead of 63 classes, the work in [1] uses a coarse grained set of 12 supersenses. Their results on recurrent models are much better. We are still trying to figure out the mapping between the fine grained 63 supersenses to these coarse grained senses. We have also mailed the authors of this paper and are waiting for a response. Hence, we aim to use the coarse grained supersenses for our further evaluation.

6. REFERENCES

[1] H. Gonen and Y. Goldberg. Semi supervised preposition-sense disambiguation using multilingual data. *arXiv preprint arXiv:1611.08813*, 2016.

[2] K. Litkowski and O. Hargraves. Semeval-2007 task 06: Word-sense disambiguation of prepositions. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 24–29. Association for Computational Linguistics, 2007.

[3] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[4] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[5] M. Sawant, T. Sangoi, and S. Nair. Supervised word sense disambiguation.

[6] N. Schneider, J. D. Hwang, V. Srikumar, M. G. A. S. K. Conger, and T. O. M. Palmer. A corpus of preposition supersenses. *LAW X*, page 99, 2016.

[7] P. Ye and T. Baldwin. Melb-yb: Preposition sense disambiguation using rich semantic features. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 241–244. Association for Computational Linguistics, 2007.

[8] D. Yuret. Ku: Word sense disambiguation by substitution. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 207–213. Association for Computational Linguistics, 2007.