CS229 Project Report: NER adaptation

Pi-Chuan Chang **Surabhi Gupta** **Yun-Hsuan Sung**

1 NER Approaches: MEMM

Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) is a kind of sequential model, which models the likelihood by a sequence of states given Markov assumption. Instead of using state transition probability and observation probability given a specific state in HMM, MEMM models the transition probability of the current state given both previous state and observation.

$$\begin{aligned} P(\underline{s}|\underline{o}) &= \sum_{i=1}^{N} P(s_i|s_1...s_{i-1}, \underline{o}) \\ &= \sum_{i=1}^{N} P(s_i|s_{i-1}, \underline{o}) \end{aligned}$$

For each transition probability, MEMM uses maximum entropy model. Maximum entropy model is used to model the data distribution, which gives us as much information as possible. Without any constraints, the uniform distribution gives us maximum entropy. However, we have training data which gives some facts about the true distribution. What maximum entropy model does is to maximize the entropy of the data give such constraints coming from the training data. The constraints are that the expected value of each feature in the distribution is the same as its actual count in the training data. I.e.

$$\sum_{i=1}^{n} f_a(\underline{o}_{t_i}, s_{t_i}) = \sum_{i=1}^{n} \sum_{s \in S} P(s|s_{i-1}, \underline{o}_{t_i}) f_a(\underline{o}_{t_i}, s)$$

,where $t_1...t_n$ are the time index such that $s_{t_i} = s_{i-1}$.

Also, the features are binary indicator function of both current observation and a possible new current state. It can be shown that the entropy model given those constraints has exponential form:

$$P(s_i|s_{i-1}, \underline{o}) = \frac{1}{Z(\underline{o}, s_{i-1})} exp(\sum_a \lambda_a f_a(\underline{o}, s_i))$$

which is the same as the distribution which maximizes likelihood.

2 Data sets

We used the datasets and evaluation scripts developed for the CoNLL NER English task (Sang and Meulder, 2003). There are three named entity classes we are interested in: LOCATION, ORGANIZATION, and PERSON[1]. Other than these, there is also a background symbol used to mark non-named entity words. The basic training set (**conll-train** is part of the Reuters Corpus, and there are $203,621$ words. The CoNLL-2003 shared task defined a development set (**conll-testa**) and a final test set(**conll-testb**). For now, the experiments we've done are using MEMM classifier.

Another data set we use is the Email datasets. Email datasets includes 3 datasets: EnronMeetings, EnronRandom and NewsGroups. In our experiments, the performance on EnronRandom sets is the highest. The reason that results on EnronRandom are better than EnronMeetings is because the EnronRandom training data (150300 words) is more than EnronMeetings training data (70067 words). However, even though NewsGroups training data has 117986 words, the result on the NewsGroups data is the worst in 3 sets. It is because the data in NewsGroups is very diverse; there are even a lot of codes inside.

Times of India data: We wanted some data that had completely different names from American news in order to test for cotraining (Section 4). We went to the Times of India website and downloaded the documents. We hand annotated 46 documents with information of Person, Location and Organization. This data was then converted to the CONLL format for use in our experiments.

3 How to do adaptation on MEMM

Adaptation is to train a model in one domain which has enough training data. Then, instead of retraining a new model in the new domain, we adapt the existing model to fit the new domain by using a small amount of training data. (Chelba and Acero, 2004) used MEMM for the task of capitalizing text by using MAP adaptation with a prior distribution on the model parameters(Chen and Rosenfeld, 2000). We use the same technique to do adaptation in Named Entity Recognition.

Apply MAP adaptation with a prior distribution to MEMM can be formulated as:

$$\begin{aligned} L(\Lambda) &= \sum_{x,y} \tilde{p}(x,y) \log(p_\Lambda(y|x)) \\ &\quad - \sum_{i=1}^{F} \frac{(\lambda_i - \lambda_i^0)^2}{2\sigma_i^2} + c(\Lambda) \end{aligned}$$

[1] In CoNLL-2003 shared task, there was a fourth class: MISC.

where $\lambda_i^0, i = 1...F$ are the parameters trained in old domain. $\lambda + i, i = 1...F$ are the parameters adapted by new domain data.

In this maximum log-likelihood formula, we add the second term in right hand side to compensate the problem of insufficient data in new domain. Instead of retraining the new parameters, we consider both the likelihood for new parameters and the distance between new parameters and old parameters. Then we optimize their sum. The second term can also be explained as adding some penality if the new parameters are too far from old parameters. To estimate the parameters in our MEMM, we used limited-memory Quasi-Newton BFGS based on the algorithms in Jorge Norcedal's Numerical Optimization Book(Nocedal, 1999).

The variance of the prior terms can also be tuned. When the variance becomes large, the regularization power of the prior becomes weaker. A more fine-grained adaptation is to adjust the covariance matrix as a diagonal matrix, where the diagonal terms are the variance of each corresponding feature. But in MEMM, the features space is sparse so it is not easy to compute the variance of each of the features.

4 Cotraining with the internal and external views

Cotraining can be used when there exist two separate views of the data, like two disjoint sets (Blum and Mitchell, 1998). The goal of cotraining is to use these two different views to leverage the results on unlabeled data. In the case of our data, we have two views - the internal view and the external view. Internal view can be thought of as the local features (the specific names used), and the external as the global features (such as the structure of the content). When we compare the say the Times of India and American news content, the external view is similar, but the internal views are different. Similarly when comparing say email and news data, the internal views are the same but the external views are different.

One strategy that has been applied by Blum and Mitchell, is to train two separate learning algorithms separately on each view, and then each of the algorithms predictions on the new unlabeled examples are used to enlarge the training set of the other. One motivation for this approach is that it is easier to get unlabeled data than labeled data. In another strategy, Sarkar(Sarkar, 2001) mentions that training a statistical parser on combined labeled and unlabeled data strongly outperforms training on only the labeled data. Nigam, Ghani (Nigam and Ghani, 2000) demonstrated that when learning from labeled and unlabeled data, algorithms can leverage an natural independent split of features if one exists or create a split, and this outperforms algorithms that do not.

Here we present a split of the features: Table 4 presents a split of the features that we have used. The notation used is as follows: s_0 - shape of current word, s_{-1} - shape of previous word, s_1 - shape of next word. w_0 - shape of current word, w_{-1} - shape of previous word, w_1 - shape of next word.

Algorithm used as adapted from (Nigam and Ghani, 2000) is described in Figure 1.

5 Random Forest

Random Forest (Breiman and Cutler,) is a set of decision trees. To classify a new piece of data, each decision tree make its own classification. The final decision is made by a vote between the decision trees and the forest chooses the label which has the most votes. It was shown in the classification that the error rate depends on the following:

1. The correlation between any two trees in the forest. Increasing the correlation increases the error rate.
2. The strength of each individual tree. Increasing the strength of each tree reduces the error rate.

Instead of doing adaptation to get a better classifier in new domain, we increase the robustness of classifier by building a random forest. In our setting, each tree in the random forest is represented by a MEMM classifier. We separate our all features into 24 different classes by their properties. Each MEMM classifier is randomly assigned some of classes of features. Unlike voting between classifiers, we add the probability of each classifier and choose the class which gets the highest probability score.

$$class = \underset{c}{argmax} \sum_i P_i(c|\underline{o})$$

The unbalanced number between the background class and named entity class causes the classifier to label the data as background class. This unbias effect became more obvious when we combined the probability score between classifiers. In order to overcome this unbias effect, we introduced a penalty threshold to background class. We label the background class until the score of background is larger than that of named entity.

6 Experimental Results

6.1 Baseline experiments: Matched cases (training and testing data from same domain)

We have done experiments of matched environments for training and testing. There are three data sets in the Email data. See Table 2, 3.

Internal	External	Ambiguous
w_0	w_{-1}	w_0, w_1
prefix and suffix of w_0	w_1	$s_{-1}s_0, s_0s_1, s_{-1}s_0s_1$
s_0	$s_{-1}, s_1, w_{-1}s_0, w_1s_0$	useTypeySequence
		(TPS(in CpC), TTPS(inCpCp2C))
co-occurrence in a window	useLastRealWord	
usePrevSequences and useSequences (PSEQ)	useNextRealWord	
useTypeySequence (shape-TPS2)	useDisjunctive	
useLongSequence (PPSEQ)	useTypeySequences (TNS1)	
	useLongSequence (PPSEQ)	

Table 1: Features separated as internal and external

```
trainingDataA, trainingDataB = training data
While there exist documents without class labels
{
       Build classifier A using the first split of features and the training
data
       Build classifier B using the second split of features and the training
data
       Go through all the unlabelled documents
       {
           Document d = one which A is most confident about
           Add d to trainingDataA
           Remove d from unlabelled data
       }
       Repeat for classifier B
}

Testing:
Use predictions of classifier A and B by multiplying together their
score on the test data.
```

Figure 1: Co-training algorithm

train:	**conll-train**			
test:	**conll-testa**			
conlleval		P	R	F
	PER	91.73	92.69	92.21

Table 2: Baseline results on the CoNLL data

train:	Email train			
test:	Email test			
datasets		P	R	F
EnronRandom	PER	82.85	88.46	85.56
EnronMeetings	PER	85.42	80.15	82.70
NewsGroups	PER	78.62	76.81	77.17

Table 3: Results on the Email data

6.2 Mismatched cases

To show how the NER performance degrades when moving to a different testing environment, See Table 4.

train:	**conll-train** 1 class			
test:	Email test			
datasets		P	R	F
EnronRandom	PER	72.76	70.33	71.53
EnronMeetings	PER	83.66	41.43	55.41
NewsGroups	PER	62.12	64.42	63.25

Table 4: Mismatched: Train on CoNLL; test on Email datasets

6.3 MAP Adaptation

Applying the adaptation approach is described in Section 3. With all the training data of each Email data set as adaptation data, we get the result in Table 5. One question is that if we have so many "adaptation" data, why can't we just combine it with the CoNLL training data and train a classifier? We also tried that (Table 6), and the result shows that doing MAP adaptation gives better performance. The MAP adaptation performance is also better than that in Table 3.

train:	**conll-train** (1 class)			
test:	Email test			
adapt:	Email train			
datasets		P	R	F
EnronRandom	PER	89.36	87.11	88.22
EnronMeetings	PER	89.05	81.50	85.11
NewsGroups	PER	80.07	76.11	78.04

Table 5: Using all the Email training data as adaptation data

Instead of using all the EnronMeetings-train data as adaptation data, we also tried to increase the amount of adaptation data from 0 documents to all of the 729 documents. We plot the F score curve as the number of documents increases. (Figure 2) In this figure, when we put in 10 or 20 documents as adaptation data, the performance dropped. Since we have features that can decide 1 to 3 labels at the same time, too few adaptation data

train: test:	**conll-train**(1 class) + Email train **Email test**			
datasets		P	R	F
EnronRandom	PER	85.83	88.46	87.12
EnronMeetings	PER	90.95	77.46	83.66
NewsGroups	PER	78.38	78.94	78.66

Table 6: Combining conll-train and Email training set as training data

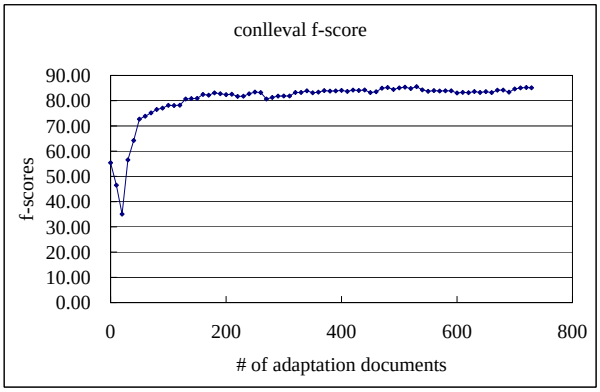

Figure 2: Adaptation F-score to the number of adaptation documents

might just hurt the performance. But the performance improves after 30 documents, and when the number of adaptation data reaches 140 (about 10,000 words), the performance is reasonably good. When using all the 729 documents to adapt, the MAP adaptation outperforms throwing in everything as training data or training on only EnronMeetings-train. In fact, using MAP adaptation as in Table 5 gives us the best performance we have on this dataset.

6.4 Cotraining

We used CoNLL as our training data and did two sets of experiments. In the first case, we tested on Times of India data and in the second case we tested on the EnronMeetings data (see table 7. In each case the unlabeled data we used is from the same domain as the test data.

According to these results it seems that cotraining did worse than the baseline of the MEMM classifier without the use of any unlabeled documents. We have some intuition about why this could be the case which we have described in Section 7.

	Overall	Person	Location	Organization
TOI Baseline	62.04	74.09	48.30	43.85
TOI Cotraining	51.15	59.86	44.65	27.75
Enron Baseline	64.38	64.38		
Enron Cotraining	39.57	39.57		

Table 7: F scores: train on conll-train; test on Enron, TOI data

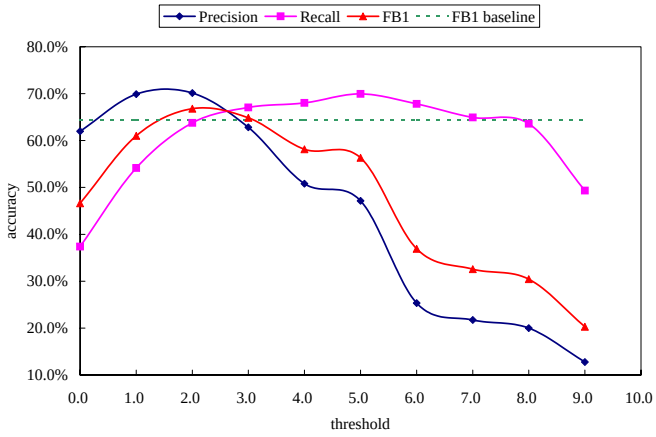

Figure 3: EnronMeetings with 10 classifiers and 16 classes of features

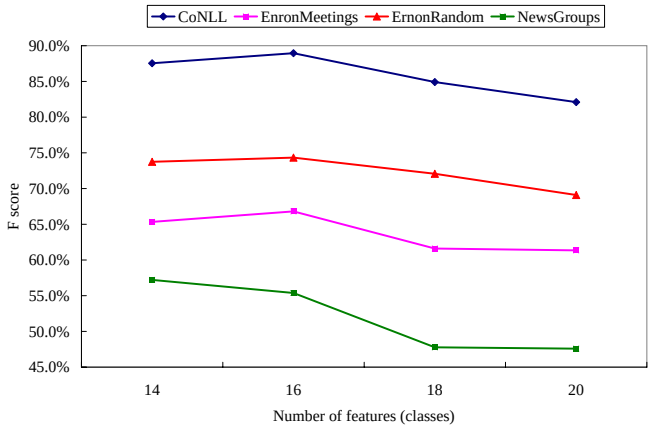

Figure 4: best result on EnronMeetings with 10 classifiers

6.5 Random Forest

In these sets of experiments, we used CoNLL training set as our training data and tested on CoNLL test set, EnronMeetings, EnronRandom, and NewsGroups. we trained our random forest with 10, 20, and 30 MEMM classifiers. We also adjusted each MEMM classifiers with randomly chosen 14, 16, 20, and 22 classes of features. In the final decision part, we tried different penalty threshold, from 0.0 9.0 to the background class.

In Figure 3, when change the penalty threshold from 0.0 to 9.0, F1 score increases, achieves maximum around 2.0, then decrease. And the maximum F1 score is 66.80%, which improves around 2%. In Figure 4, using more classes of features in random forest decreases the F1 score in all the test set. Besides, we have maximum F1 score 88.95%, 66.80%, and 74.33% by using 16 classes of features in CoNLL, EnronMeetings, and EnronRandom, respectively. For NewsGroups, we get maximum F1 score 57.21% by using 14 classes of features

	Overall	Person
Enron Baseline	36.92	36.92
Enron Cotraining	39.57	39.57

Table 8: F scores: train on conll-train; test on Enron data

7 Analysis

7.1 MAP Adaptation

MAP adaptation on the mean of the Gaussian gives us better performance on each of the 3 email data sets compared to only training on conll data.

7.2 Co-training

Co-training does not seem to work well for our task. We manually created a split of features into internal and external. It seems that does not work with the current split of features. When we trained our MEMM classifier on only one set of features, then co-training outperformed the experiments without cotraining as in Table 8. This is something that we would like to further explore in the future.

7.3 Random Forest

Using random forest in NER increases the robustness a little bit in some new domain. We get around $2 \sim 3\%$ gain in both EnronMeetings and EnronRandom. However, we get worse result in Newsgroups and also compensate the performance in original domain.

In Figure 3, the precision doesn't monotonically decrease and the recall doesn't monotonically increase when we increase the penalty to background class. The reason is that in the standard CoNLL evaluation, it adds positions, B and I to the label, which means beginning and intermediate. The result is correct only when both the label and position are correct. Although increasing the threshold, we tends to label more aggressively to non-background class. It is still possible to get wrong position then get wrong result.

8 Acknowledgments

We would like to thank Prof. Chris Manning for his help with this project. We consulted him for most of our issues with our implementation on Named Entity Recognition and he gave us valuable feedback. We would also like to thank Prof Andrew Ng whom we consulted for specific machine learning algorithms. We learnt a lot from the lecture on Debugging Learning Algorithms.

9 Future Work

We will try out the cotraining algorithm with a different split of features, because our results showed the current features split actually hurt the performance too much. We will also try to use CRF classifier instead of MEMM classifier to see if we can get more improvements. Also, we can include features from parser as a knowledge of recognizing the named entities. And we're also interested in using unlabelled data to get useful features of NER adaptation.

References

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *COLT: Proceedings of the Workshop on Computational Learning Theory, Morgan Kaufmann Publishers*, pages 92–100.

Leo Breiman and Adele Cutler. Random forest. www.stat.berkeley.edu/users/breiman/RandomForests.

Ciprian. Chelba and Alex Acero. 2004. Adaptation of maximum entropy capitalizer: Little data can help a lot. In *Proccedings of EMNLP*.

Stanley F. Chen and Ronald Rosenfeld. 2000. A survey of smoothing techniques for me models. In *IEEE Transactions on Speech and Audio Processing*.

Andrew McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy Markov models for information extraction and segmentation. In *Proc. 17th International Conf. on Machine Learning*, pages 591–598. Morgan Kaufmann, San Francisco, CA.

Kamal Nigam and Rayid Ghani. 2000. Analyzing the effectiveness and applicability of co-training. In *CIKM*, pages 86–93.

Jorge Nocedal. 1999. *Numerical Optimization*. Springer-Verlag New York.

Eric F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 2003 Conference on Computational Natural Language Learning*.

Anoop Sarkar. 2001. Applying co-training methods to statistical parsing. In *Proceedings of the 2nd Meeting of the North American Association for Computational Linguistics: NAACL 2001*, pages 175–182.