

# Adaptive parameters for entity recognition with perceptron HMMs

Massimiliano Ciaramita\*

Google  
Zürich, Switzerland  
massi@google.com

Olivier Chapelle

Yahoo! Research  
Sunnyvale, CA, USA  
chap@yahoo-inc.com

## Abstract

We discuss the problem of model adaptation for the task of named entity recognition with respect to the variation of label distributions in data from different domains. We investigate an adaptive extension of the sequence perceptron, where the adaptive component includes parameters estimated from unlabelled data in combination with background knowledge in the form of gazetteers. We apply this idea empirically on adaptation experiments involving two newswire datasets from different domains and compare with other popular methods such as self training and structural correspondence learning.

## 1 Introduction

Model adaptation is a central problem in learning-based natural language processing. In the typical setting a model is trained on annotated *in domain*, or *source*, data, and is used on *out of domain*, or *target*, data. The main difference with respect to similar problems such as semi-supervised learning is that source and target data are not assumed to be drawn from the same distribution, which might actually differ in relevant distributional properties: topic, domain, genre, style, etc. In some formulations of the problem a few target labeled data is assumed to be available (Daumé III, 2007). However, we are interested in the case in which no labeled data is available from the target domain – except for evaluation purposes and fine tuning of hyperparameters.

Most of the work in adaptation has focused so far on the input side; e.g, proposing solutions based on generating shared source-target representations (Blitzer et al., 2006). Here we focus instead on the output aspect. We hypothesize that

---

This work was carried out while the first author was working at Yahoo! Research Barcelona.

part of the loss incurred in using a model out of domain is due to its built-in class priors which do not match the class distribution in the target data. Thus we attempt to explicitly correct the prediction of a pre-trained model for a given label by taking into account a noisy estimate of the label frequency in the target data. The correction is carried out by means of adaptive parameters, estimated from unlabelled target data and background “world knowledge” in the form of gazetteers, and taken in consideration in the decoding phase. We built a suitable dataset for experimenting with different adaptation approaches for named entity recognition (NER). The main findings from our experiments are as follows. First, the problem is challenging and only marginal improvements are possible under all evaluated frameworks. Second, we found that our method compares well with current state-of-the-art approaches such as self training and structural correspondence learning (McClosky et al., 2006; Blitzer et al., 2006) and taps on an interesting aspect which seems worth of further research. Although we concentrate on a segmentation task within a specific framework, the perceptron HMM introduced by Collins (2002), we speculate that the same intuition could be straightforwardly applied in other learning frameworks (e.g., Support Vector Machines) and different tasks (e.g., standard classification).

## 2 Related work

Recent work in domain adaptation has focused on approaches such as *self-training* and *structural correspondence learning* (SCL). The former approach involves adding self-labeled data from the target domain produced by a model trained in-domain (McClosky et al., 2006). The latter approach focuses on ways of generating shared source-target representations based on good cross-domain (pivot) features (Blitzer et al., 2006) (see

also (Ando, 2004)). Self training has proved effective in syntactic parsing, particularly in tandem with discriminative re-ranking (Charniak and Johnson, 2005), while the SCL has been applied successfully to tasks such as PoS tagging and opinion analysis (Blitzer et al., 2006; Blitzer et al., 2007). We address a different aspect of the adaptation problem, namely the difference in label distributions between source and target domains. Chan and Ng (2006) proposed correcting the class priors for domain adaptation purposes in a word sense disambiguation task. They adopt a generative framework where the base model is a naive Bayes classifier and priors are re-estimated with EM. The approach proposed by Chelba and Acero (2004) is also related as they propose a MAP adaptation via Gaussian priors of a MaxEnt model for recovering the correct capitalization of text.

Domain adaptation naturally invokes the existence of a specific task and data. As such it is natural to consider the modeling aspects within the context of a specific application. Here we focus on the problem of named entity recognition (NER). There is still little work on adaptation for NER. Ando (2004) reports successful experiments on adapting with an SCL-like approach, while Ciaramita and Altun (2005) effectively used external knowledge in the form of gazetteers in a semi-Markov model. Mika *et al.* (2008) used Wikipedia to generate additional training data for domain adaptation purposes.

### 3 Problem statement

Named entity taggers detect mentions of instances of pre-defined categories such as person (Per), location (Loc), organization (Org) and miscellaneous (Misc). The problem can be naturally framed as a segmentation and labeling task. State of the art systems, e.g., based on sequential optimization, achieve excellent accuracy in domain. However, accuracy degrades if the target data diverges in relevant distributional aspects from the source. As an example, the following is the output of a perceptron HMM<sup>1</sup> trained on the CoNLL 2003 English data (news) (Sang and Muelder, 2003) when applied to a molecular biology text:<sup>2</sup>

<sup>1</sup>We used the implementation available from <http://sourceforge.net/projects/supersensetag>, more details on this tagger can be found in (Ciaramita and Altun, 2006).

<sup>2</sup>The same model achieves F-scores well in excess of 90% evaluated in domain.

- (1) Cdc2-cyclin<sup>Org</sup> B-activated Polo-like<sup>Misc</sup>  
kinase specifically phosphorylates at least three  
components of APC<sup>Org</sup>.

The tagger predicts several CoNLL entities which are unlikely to occur in that context. One source of confusion is probably the shape of words, including case, numbers, and non alphabetical characters, which are also typical, and thus misleading, of unrelated CoNLL entities. However, we argue that the problem is partially due to the parameters learned which reflect the distribution of classes in the source data. The parameter, acting as biased priors, lead the tagger to generate inappropriate distributions of labels. We propose that this aspect of the problem might be alleviated by correcting the score for each class with an estimate of the class frequency in the target data. Thus, with respect to the example, we would like to decrease the score of “Org” labels according to their expected frequency in a molecular biology corpus.

### 4 A perceptron with adjustable priors

As generic taggers we adopt perceptron-trained HMMs (Collins, 2002) which have excellent efficiency/performance trade-off (Nguyen and Guo, 2007). The objective of learning is a discriminant  $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ , where  $\mathcal{Y}$  denotes sequences of labels from a pre-defined set of categories  $Y$ .  $F(\mathbf{x}, \mathbf{y}; \alpha) = \langle \alpha, \Phi(\mathbf{x}, \mathbf{y}) \rangle$  is linear in a feature representation  $\Phi$  defined over a joint input/output space,<sup>3</sup> a global feature representation mapping each  $(\mathbf{x}, \mathbf{y})$  pair to a vector of feature counts  $\Phi(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d$ :

$$[\Phi(\mathbf{x}, \mathbf{y})]_i = \sum_{j=1}^{|\mathbf{y}|} \phi_i(y_{j-1}, y_j, \mathbf{x}), \quad (2)$$

where  $\phi_i$  is a (binary) predicate. Given an input sequence  $\mathbf{x}$ , we find the optimal label sequence,  $f(\mathbf{x}; \alpha) = \arg \max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}; \alpha)$ , with Viterbi decoding. The model  $\alpha$  is learned with the perceptron algorithm.

Each feature represents a spelling or contextual property, or the previous label. The simplest baseline (model B) uses the features listed in the upper half of Table 1. In previous work on NER adaptation, Ciaramita and Altun (2005) found that gazetteers, in combination with semi-Markov models, significantly improved adaptation. Similarly, we define additional features using

<sup>3</sup> $\langle \mathbf{u}, \mathbf{v} \rangle$  denoting the inner product between  $\mathbf{u}$  and  $\mathbf{v}$ .

Model B features			
Feature	example token	feature value(s)	Position
Lowercase word	Pierre	pierre	$i-1, i, i+1$
Part of Speech	Pierre	NNP	$i-1, i, i+1$
Word Shape	Pierre	Xx	$i-1, i, i+1$
Suffix <sub>2/3</sub>	Pierre	{re, rre}	$i$
Prefix <sub>2/3</sub>	Pierre	{pi, pie}	$i$
Previous label	Vinken (in “Pierre Vinken”)	B-PER (label on “Pierre”)	$i$
Additional features of model BG			
Feature	example token	feature value(s)	Position
InGazetteer	Islands (in “Cayman Islands”)	I-Country <sub>2</sub> (inside a 2-word country name)	$i-1, i, i+1$
Most frequent supersense	Eve	B-Per <sub>1</sub> (1 token Person label)	$i$
2 most frequent supersenses	Eve	B-Per-Time <sub>1</sub> (1 token Person/Time label)	$i$
Number of supersenses	Eve	B-NSS <sub>4</sub> <sub>1</sub>	$i$

**Table 1.** Feature list and examples. The upper half lists the features for the baseline tagger (B), the lower half lists the additional features extracted from the gazetteers included to the second non-adapted tagger (BG). The last number on the feature indicates the length of the entry in the list; e.g., “Islands” in the example is the end of a two-word item, in the country gazetteer, because of “Cayman Islands”. The remaining features capture the most frequent Wordnet supersense of the word, the first and second most frequent supersenses, and the total number of supersenses.

the gazetteers from GATE,<sup>4</sup> (Cunningham et al., 2002) namely, countries, person first/last names, trigger words; and also from Wordnet: using the lexicographers or *supersense* labels; and a list of company names from Fortune 500. For this second baseline (model BG) we also extract the features in the bottom half of Table 1.

#### 4.1 Decoding with external priors

In our method training is performed on the source data using the perceptron algorithm. Adaptation takes place at decoding time, when the score of the entity labels is adjusted according to a  $k$ -dimensional parameter vector  $\theta$ ,  $k = |Y|$ , estimated by comparing the source and the unlabeled target data. The score of a sequence  $\hat{y}$  for input  $\mathbf{x}$  in the target domain is computed with a variant of the original discriminant:

$$F'(\mathbf{x}, \mathbf{y}; \alpha) = \sum_{j=1}^{|\mathbf{y}|} \left( \sum_{i=1}^d \phi_i(y_{j-1}, y_j, \mathbf{x}) \alpha_i \right) + \tau \theta_{y_j} \quad (3)$$

where  $\theta_{y_j}$  is the adaptive parameter associated with  $y_j$ , and  $\tau$  is a scaling factor. The new prediction for  $\mathbf{x}$  is  $f'(\mathbf{x}; \alpha) = \arg \max_{\mathbf{y} \in \mathcal{Y}} F'(\mathbf{x}, \mathbf{y}; \alpha)$ .

## 5 Adaptive parameters

### 5.1 Theta

The vector  $\theta$  encodes information about the expected difference in frequency of each category between source and target. Let  $g_Q(c) =$

$\frac{\text{count}(c, Q)}{\sum_{c'} \text{count}(c', Q)}$  be an estimate of the relative frequency of class  $c$  in corpus  $Q$ . We propose to formulate  $\theta_c$  as:

$$\theta_c = \frac{g_T(c) - g_S(c)}{g_S(c)} \quad (4)$$

where  $T$  and  $S$  are, respectively, the source and target data. This is the ratio of the difference between in and out domain relative frequencies for class  $c$ , with respect to the in domain frequency. Intuitively,  $g_S(c)$  represents an estimate of the frequency of  $c$  in the source  $S$ , and  $\theta_c$  an estimate of the expected decrease/increase as a fraction of the initial guess;  $\theta_c$  is negative if class  $c$  is less frequent in the target data than in the source data, and positive otherwise. From this, it is clear that equation (3) will offset the scores in the desired direction.

A crucial issue is the estimation of  $\text{count}(c, Q)$ , a guess of the frequency of  $c$  in  $Q$ . A simple solution could be to count directly the class frequencies from the labeled source data, and to obtain a noisy estimate on the target data by counting the occurrence of entities that have known labels in the source data. This approach unfortunately works very badly for at least two reasons. First, the number of entities in each class reflects the frequency of the class in the source. Therefore using lists of entities from the source as proxies for the class in the target data can transfer the source bias to the target. Second, entities can have different senses in different domains; e.g., several English city names occur in the Wall Street Journal as locations (Liverpool, Manchester, etc.) and

<sup>4</sup><http://www.gate.ac.uk/>.

Attribute	CoNLL	BBN-4
# tokens	300K	1.046M
Source	Reuters	Wall Street Journal
Domain	General news	Financial
Years	1992	1987
# entities	34,841	58,637
Loc	30.48%	22.51%
Per	28.58%	20.08%
Org	26.55%	46.27%
Misc	14.38%	10.41%

**Table 2.** BBN and CoNLL datasets.

in Reuters news as both locations and organizations (football clubs). We propose to use lists of words which are strongly associated with entities of specific classes but are extracted from an independent third source. In this way, we hope the bias they carry will be transferred in similar ways to both source and target. Similarly, potential ambiguities should be randomly distributed between source and target. Thus, as a first approximation, we propose that given a list of words  $L_c$ , supposedly related to  $c$  and generated independently from source and target,  $\text{count}(c, Q)$  can be defined as:

$$\text{count}(c, Q) \equiv \sum_{w \in L_c} \text{count}(w, Q) \quad (5)$$

## 5.2 Tau

The scalar  $\tau$  needs to be large enough to revise the decision of the base model, if necessary. However,  $\tau$  should not be too large, otherwise the best prediction of the base model would be ignored. In order for  $\tau$  to have an effective, but balanced, magnitude we introduce a simple notion of margin. Let the score of a given label  $y_s$  on token  $s$  be:  $G(\mathbf{x}, y_s; \alpha) = \sum_{i=1}^d \phi_i(y_{s-1}, y_s, \mathbf{x}) \alpha_i$ , and let  $\hat{y}_s = \arg \max_{y \in \mathcal{Y}} G(\mathbf{x}, y; \alpha)$ , we define the margin on  $s$  as:

$$M_s \equiv \min_{y_s \neq \hat{y}_s} (G(\mathbf{x}, \hat{y}_s; \alpha) - G(\mathbf{x}, y_s; \alpha)). \quad (6)$$

The mean of  $M$  provides a rough quantification of the necessary amount by which we need to offset the scores  $G(\mathbf{x}, y_s; \alpha)$  in order to change the predictions. As a first guess, we take  $\tau = \mu(M_S) = \frac{1}{|S|} \sum_s^{S} M_s$ , which we interpret as an upper bound on the desired value of  $\tau$ . While experimenting on the development data we found that  $\tau/2$  yields good results.

## 6 Experimental setup

### 6.1 Data

We used two datasets for evaluation. The first is the English CoNLL 2003 dataset (Sang and Mueelder, 2003), a corpus of Reuters news annotated with person, location, organization and miscellaneous entity tags. The second is the BBN corpus (BBN, 2005), which supplements the WSJ Penn TreeBank with annotation for 105 categories: named entities, nominal entities and numeric types. We made the two datasets “semantically” compatible as follows. We tagged a large collection of text from the English Wikipedia with CoNLL and BBN taggers. We counted the frequencies of BBN/CoNLL tag pairs for the same strings, and assigned each BBN tag the most frequent CoNLL tag;<sup>5</sup> e.g.,

BBN tag	CoNLL tag
Work_of_art:Book	→ Misc
Organization:Educational	→ Org
Location:Continent	→ Loc
Person	→ Per

48 BBN-to-CoNLL pairs were labelled in this way. Remaining categories, e.g., descriptive and numerical types, were mapped to the Outside tag as they are not marked in CoNLL. Finally, we substituted all tags in the BBN corpus with the corresponding CoNLL tag, we call this corpus BBN-4. The data is summarized in Table 2. Notice the different label distributions: the BBN-4 data is characterized by a skewed distribution of labels with organization by far the most frequent class, while the CoNLL data has a more uniform distribution with location as the most frequent class. The CoNLL data was randomly split in three disjoint sets of sentences for training (16,540 sentences), development (2,068) and test (2,136). For BBN-4 we used WSJ sections 2-21 for training (39,823), section 22 for development (1,700) and section 23 for test (2,416). We evaluated models in both directions; i.e., swapping CoNLL and BBN-4 as source/target.

### 6.2 Model tuning

We regularize the perceptrons by averaging (Freund and Schapire, 1999). The perceptron HMM

<sup>5</sup>A simpler approach might that of manually mapping the two tagsets, however a number of cases that are not trivial to resolve emerges in this way. For this reason we decided to adopt the described data-driven heuristic approach.

has only one hyper-parameter, the number of training iterations (or epochs). Models trained for application out of domain can benefit from early stopping which provides an additional mean of regularization. For all models compared we used the development sets for choosing the number of epochs for training the perceptron on the source data. This is an important step as different adaptation approaches yield different overfitting pattern and it is important to control for this factor for a fair comparison. As an example, we found that the self-training models consistently overfit after just a few iterations after which performance has a steep drop. The order of presentation of instances in the training algorithm is randomized; for each method we repeat the process 10 times and report average F-score and standard error.

The vector  $\theta$  was estimated using one of the same gazetteers used in the base tagger (BG), a list of 1,438 *trigger words* from GATE.<sup>6</sup> These are words associated with certain categories; e.g., “abbess/Per”, “academy/Org”, “caves/Loc”, and “manifesto/Misc”. The lists for different classes contain varying numbers of items and might contain misleading words. To obtain more reliable estimates of comparable magnitude between classes we computed equation (4) several times by sampling an equal number of words from each list and taking the mean. On the development set this proved better than computing the counts from the entire list.

Other sources could be evaluated, for example lists of entities of each class extracted from Wikipedia. We used all single-word triggers: 191 for Loc, 171 for Misc, 89 for Org and 592 for Per. With each list we estimated  $\theta$  as in Section 5.1 for each of the four labels starting with “B”, i.e., entity beginnings,  $\theta = 0$  for the other five labels. To find  $\theta$  we use as source  $S$ , the in-domain data, and as target  $T$  the out-domain data. The lists contain different number of items and might contain misleading words.

To set  $\tau$  we compute the mean margin (6) on CoNLL, using the tagger trained on CoNLL ( $\text{mean}(M_s) \approx 50$ ), similarly for BBN-4 ( $\text{mean}(M_s) \approx 38$ ). We used the development set to fine tune the adaptive rate setting it equal to  $\tau = \frac{1}{2}\text{mean}(M_s)$ .

<sup>6</sup>This list corresponds to the list of words  $L_c$  of Section 5.1.

### 6.3 Self training

To compare with self-training we trained a tagger (BG) on the training set of CoNLL. With the tagger we annotated the training set of BBN-4, and added the self-labeled data, 39,823 BBN-4 sentences, to the gold standard CoNLL training. Similarly, in the reverse direction we trained a tagger (BG) on the training set of BBN-4, annotated the training set of CoNLL, and added the self-labeled 16,540 CoNLL sentences to the BBN-4 training. We denote these models  $\text{BG}_{SELF}$ , and the augmented sources as CoNLL+ and BBN-4+.

### 6.4 Structural correspondence learning

We first implemented a simple baseline following the idea presented in (Ando, 2004). The basic idea consists in performing an SVD decomposition of the feature-token matrix, where the matrix contains all the sentences from the source and target domains. The goal is to capture co-occurrences of features and derive new features which are more stable. More specifically, we extracted the 50 principal directions of the feature-token matrix and projected all the data onto these directions. This results in 50 new additional features for each token that we append to the original (sparse binary) feature vector  $\phi_i$ ,  $1 \leq i \leq d$ . In order to give equal importance to the original and new features, we multiplied the new features by a constant factor such that the average  $L_1$  norms of the new and old features are the same. Note that this weighting might not be optimal but should be sufficient to detect if these new features are helpful or not.

We then implemented several versions of structural correspondence learning. First, following the original formulation (we refer to this model as SCL1), 100 *pivot* features are selected, these are frequent features in both source and target data. For a given pivot feature  $k$ , a vector  $\mathbf{w}_k \in \mathbb{R}^d$  is computed by performing a regularized linear regression between all the other features and the given pivot feature. The matrix  $W$  whose columns are the  $\mathbf{w}_k$  is formed and the original feature vectors are projected onto the 50 top left singular vectors of  $W$ , yielding 50 new features. We also tried the following variants. In the version we refer to as SCL2 we rescale the left singular vectors of  $W$  by their corresponding singular values. In the last variant (SCL3) we select the pivot features which are frequent in the source and target domains *and* which are also predictive for the task (as measured

Model	Source	Target	Test
B	BBN-4	CoNLL	60.4 $\pm$ .28
BG	BBN-4	CoNLL	66.1 $\pm$ .32
BG <sub>SVD</sub>	BBN-4	CoNLL	66.5 $\pm$ .26
BG <sub>SCL1</sub>	BBN-4	CoNLL	66.8 $\pm$ .18
BG <sub>SCL2</sub>	BBN-4	CoNLL	64.7 $\pm$ .24
BG <sub>SCL3</sub>	BBN-4	CoNLL	66.8 $\pm$ .27
BG <sub>SELF</sub>	BBN-4+	CoNLL	65.5 $\pm$ .26
BG $_{\theta}$	BBN-4	CoNLL	66.8 $\pm$ .53

  

Model	Source	Target	Test
B	CoNLL	BBN-4	65.0 $\pm$ .77
BG	CoNLL	BBN-4	67.6 $\pm$ .69
BG <sub>SVD</sub>	CoNLL	BBN-4	67.9 $\pm$ .54
BG <sub>SCL1</sub>	CoNLL	BBN-4	67.9 $\pm$ .45
BG <sub>SCL2</sub>	CoNLL	BBN-4	68.1 $\pm$ .53
BG <sub>SCL3</sub>	CoNLL	BBN-4	67.8 $\pm$ .34
BG <sub>SELF</sub>	CoNLL+	BBN-4	68.3 $\pm$ .36
BG $_{\theta}$	CoNLL	BBN-4	70.3 $\pm$ .61

**Table 3.** Results of baselines and adaptive models.

by the mutual information between the feature and the class label). The 50 additional features are appended to the original (sparse binary) feature vector  $\phi_i$ ,  $1 \leq i \leq d$ , and again, they are first rescaled in order to have the same average  $L_1$  norm as the old features over the entire dataset.

## 7 Results and discussion

Table 3 summarizes the experimental results on both datasets. We refer to our adaptive model as BG $_{\theta}$ . Adapting a model from BBN-4 to CoNLL, self training (BG<sub>SELF</sub>, 65.5%) performs slightly worse than the base model (BG, 66.1%). The best SCL model, the original formulation, produces a small, but likely significant, improvement (BG<sub>SCL1</sub>, 66.8%). Our model (BG $_{\theta}$ , 66.8%), achieves the same result but with larger variance. The improvement of the best models over the first baseline (B, 60.4%) is considerable, +6.4%, but mostly due to gazetteers.

In the adaptation experiments from CoNLL to BBN-4 both self training (BG<sub>SELF</sub>, 68.3%) and the best SCL model (BG<sub>SCL1</sub>, 68.1%) are comparable to the baseline (BG, 67.6%). The adaptive perceptron HMM (BG $_{\theta}$ , 70.3%) improves by 2.7%, as much as model BG over B, again with a slightly larger variance. It is not clear why other methods do not improve as much. Speculatively, although we implemented several variants, SCL might benefit from further tuning as it involves several pre-processing steps. As for self training, the base tagger might be too inaccurate to support this technique. It is fair to assume that the additional hyperparameters available to our model, e.g.,  $\tau$ , provided some additional flexibility. We

also experimented with a few variants of estimating  $\theta$  on the development set; i.e., different splits of the unlabeled source/target data and different sampling modes: with and without replacement, number of trials. All of these aspects can have a significant impact on the quality of the model. This point brings up a more general issue with the type of approach explored here: while adapting the class priors seems easier than adapting the full model it is not trivial to encode noisy world knowledge into meaningful priors. Alternatively, in the presence of some labeled data one could optimize  $\theta$  directly. This information could be also elicited from domain experts. Another interesting alternative is the unsupervised estimation via EM as in (Chan and Ng, 2006).

Overall, adaptation from BBN-4 to CoNLL is harder than from CoNLL to BBN-4. A possible explanation is that adapting from specific to general is harder than in the opposite direction: the specific corpus is more heavily biased towards a domain (finance). This intuition is compatible with the baselines performing better in the CoNLL to BBN-4 direction. However, the opposite argument, that adapting from specific to general should be easier, has some appeal as well; e.g., if more general means higher entropy it seems easier to make a distribution more uniform than finding the right peak.

In general, all adaptive techniques we evaluated provided only marginal improvements over the baseline (BG) model. To put things in context, it is useful to recall that when evaluated in domain the CoNLL and BBN-4 taggers (model BG) achieve, respectively, 92.7% and 91.6% average F-scores on the test data. As the results illustrate there is a considerable drop in out domain accuracy, significantly alleviated by adding features from gazetteers and to some extent by other methods. Following Dredze *et al.* (2007) we hypothesize that a significant fraction of the loss is due to labeling inconsistencies between datasets. Although we did our best to optimize the benchmark methods it is possible that even better results could be achieved with self-training and SCL. However we stress that different methods get at different aspects of the problem: self-training targets data sparseness, SCL methods aims at generating better shared input representations, while our approach focuses on generating output distribution more compatible with the target data. It seems reason-

able to expect that better adaptation performance would result from composite approaches, aiming at both better machine learning and task-specific aspects for the named entity recognition problem.

## 8 Conclusion

We investigated the model adaptation problem for named entity recognition where the base model is a discriminatively trained HMM (Collins, 2002). We hypothesized that part of the loss incurred in using a pre-trained model out of domain is due to its built-in class priors which do not match the class distribution of the out of domain data. To test this hypothesis, and attempt a solution, we propose to explicitly correct the prediction of the model for a given label by taking into account a noisy estimate of the label frequency in the target data. We found encouraging results from preliminary experiments. It might thus be worth investigating more principled formulations of this type of method, in particular to eliminate some heuristic aspects, improve unsupervised estimations, and generalize to other classification tasks beyond NER.

## Acknowledgments

We would like to thank the anonymous reviewers for useful comments and pointers to related work.

## References

- Rie Kubota Ando. 2004. Exploiting unannotated corpora for tagging and chunking. In *Proceedings of ACL 2004*. Association for Computational Linguistics.
- BBN. 2005. Pronoun coreference and entity type corpus. *Linguistic Data Consortium (LDC) catalog number LDC2005T33*.
- John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of EMNLP 2006*. Association for Computational Linguistics.
- John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes, and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL 2007*.
- Yee Seng Chan and Hwee Tou Ng. 2006. Estimating class priors in domain adaptation for word sense disambiguation. In *Proceedings of Coling-ACL*, pages 89–96. Association for Computational Linguistics.
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL 2005*, pages 173–180. Association for Computational Linguistics.
- Ciprian Chelba and Alex Acero. 2004. Adaptation of maximum entropy capitalizer: Little data can help a lot. In *Proceedings of EMNLP*, pages 285–292. Association for Computational Linguistics.
- Massimiliano Ciaramita and Yasemin Altun. 2005. Named-entity recognition in novel domains with external lexical knowledge. In *Advances in Structured Learning for Text and Speech Processing (NIPS 2005)*.
- Massimiliano Ciaramita and Yasemin Altun. 2006. Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger. In *Proceedings of EMNLP*, pages 594–602. Association for Computational Linguistics.
- Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP 2002*, pages 1–8.
- Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of ACL 2002*. Association for Computational Linguistics.
- Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of ACL*. Association for Computational Linguistics.
- Mark Dredze, John Blitzer, Pratha Pratim Talukdar, Kuzman Ganchev, Joao Graca, and Fernando Pereira. 2007. Frustratingly hard domain adaptation for parsing. In *Proceedings of CoNLL Shared Task 2007*. Association for Computational Linguistics.
- Y. Freund and R.E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–296.
- David McClosky, Eugene Charniak, and Mark Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of COLING-ACL 2006*, pages 337–344. Association for Computational Linguistics.
- Peter Mika, Massimiliano Ciaramita, Hugo Zaragoza, and Jordi Atserias. 2008. Learning to tag and tagging to learn: A case study on Wikipedia. *IEEE Intelligent Systems*, 23(5):26–33.
- Nam Nguyen and Yunsong Guo. 2007. Comparison of sequence labeling algorithms and extensions. In *Proceedings of ICML 2007*, pages 681–688.
- Erik F. Tjong Kim Sang and Fien De Muelder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL 2003 Shared Task*, pages 142–147. Association for Computational Linguistics.