Exploiting Partial Annotations with EM Training

Dirk Hovy, Eduard Hovy
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Marina del Rey, CA 90292

{dirkh, hovy}@isi.edu

Abstract

For many NLP tasks, EM-trained HMMs are the common models. However, in order to escape local maxima and find the best model, we need to start with a good initial model. Researchers suggested repeated random restarts or constraints that guide the model evolution. Neither approach is ideal. Restarts are time-intensive, and most constraint-based approaches require serious re-engineering or external solvers. In this paper we measure the effectiveness of very limited initial constraints: specifically, annotations of a small number of words in the training data. We vary the amount and distribution of initial partial annotations, and compare the results to unsupervised and supervised approaches. We find that partial annotations improve accuracy and can reduce the need for random restarts, which speeds up training time considerably.

1 Introduction

While supervised learning methods achieve good performance in many NLP tasks, they are incapable of dealing with missing annotations. For most new problems, however, missing data is the norm, which makes it impossible to train supervised models. Unsupervised learning techniques can make use of unannotated data and are thus well-suited for these problems.

For sequential labeling tasks (POS-tagging, NE-recognition), EM-trained HMMs are the most common unsupervised model. However, running vanilla forward-backward-EM leads to mediocre results, due to various properties of the training method (Johnson, 2007). Running repeated restarts with random initialization can help escape local maxima, but in order to find the global optimum, we need to run a great number (100 or more) of them (Ravi and Knight, 2009; Hovy et al., 2011). However, there is another solution. Various papers have shown that the inclusion of some knowledge greatly enhances performance of unsupervised systems. They introduce constraints on the initial model and the parameters. This directs the learning algorithm towards a better parameter configuration. Types of constraints include ILP-based methods (Chang et al., 2007; Chang et al., 2008; Ravi and Knight, 2009), and posterior regularization (Graça et al., 2007; Ganchev et al., 2010). While those approaches are powerful and yield good results, they require us to reformulate the constraints in a certain language, and either use an external solver, or re-design parts of the maximization step. This is time-consuming and requires a certain expertise.

One of the most natural ways of providing constraints is to annotate a small amount of data. This can either be done manually, or via simple heuristics, for example, if some words' parts of speech are unambiguous. This can significantly speed up learning and improve accuracy of the learned models. These *partial annotations* are a common technique for semi-supervised learning. It requires no changes to the general framework, or the use of external solvers.

While this well-known, it is unclear exactly how much annotation, and annotation of what, is most effective to improve accuracy. To our knowledge, no paper has investigated this aspect empirically. We

NAACL-HLT Workshop on the Induction of Linguistic Structure, pages 31–38,
Montréal, Canada, June 3-8, 2012.

Inputs:	I went to the show walk on water
Partial Annotations:	I went to the*:DET* show*:NN* walk *on:sense*$_5$ water

Figure 1: In partial annotation, words are replaced by their label

explore the use of more unlabeled data vs. partial annotation of a small percentage. For the second case, we investigate how much annotation we need to achieve a particular accuracy, and what the best distribution of labels is. We test our approach on a POS-tagging and word sense disambiguation task for prepositions.

We find that using partial annotations improves accuracy and reduces the effect of random restarts. This indicates that the same accuracy can be reached with fewer restarts, which speeds up training time considerably.

Our contributions are:

- we show how to include partial annotations in EM training via parameter tying
- we show how the amounts and distribution of partial annotations influence accuracy
- we evaluate our method on an existing data set, comparing to both supervised and unsupervised methods on two tasks

2 Preliminaries

2.1 Partial Annotations

When training probabilistic models, more constraints generally lead to improved accuracy. The more knowledge we can bring to bear, the more we constrain the number of potential label sequences the training algorithm has to consider. They also help us to find a good initial model: it has to explain those fixed cases.

The purest form of unsupervised learning assumes the complete lack of annotation. However, in many cases, we can use prior knowledge to label words in context based on heuristics. It is usually not the case that all labels apply to all observations. If we know the alphabet of labels we use, we often also know which labels are applicable to which observations. This is encoded in a *dictionary*. For POS-tagging, it narrows the possible tags for each word–irrespective of context–down to a manageable set. Merialdo (1994) showed how the amount of available dictionary information is correlated with performance. However, dictionaries list all applicable labels per word, regardless of context. We can often restrict the applicable label for an observation in a specific context even more. We extend this to include constraints applied to some, but not all instances. This allows us to restrict the choice for an observation to one label. We substitute the word in case by a special token with just one label. Based on simple heuristics, we can annotate individual words in the training data with their label. For example, we can assume that "the" is always a determiner. This is a unigram constraint. We can expand those constraints to include a wider context. In a sentence like "I went to the show", we know that NN is the only applicable tag for "show", even if a dictionary lists the possible tags NN and VB. In fact, we can make that assumption for all words with a possible POS tag of NN that follow "the". This is an n-gram constraint.

Partial annotations provide local constraints. They arise from a number of different cases:

- simple heuristics that allow the disambiguation of some words in context (such as words after "the" being nouns)
- when we can leverage annotated data from a different task
- manual labeling of a few instances

While the technique is mainly useful for problems where only few labeled examples are available, we make use of a corpus of annotated data. This allows us to control the effect of the amount and type of annotated data on accuracy.

We evaluate the impact of partial annotations on two tasks: preposition sense disambiguation and POS tagging.

2.2 Preposition Sense Disambiguation

Prepositions are ubiquitous and highly ambiguous. Disambiguating prepositions is thus a challenging and interesting task in itself (see SemEval 2007 task,

(Litkowski and Hargraves, 2007)). There are three elements in the syntactic structure of prepositional phrases, namely the head word h (usually a noun, verb, or adjective), the preposition p, and the object of the preposition, o. The triple (h, p, o) forms a syntactically and semantically constrained structure. This structure is reflected in dependency parses as a common construction.

Tratz and Hovy (2009) show how to use the dependency structure to solve it. Their method outperformed the previous state-of-the-art (which used a window-based approach) by a significant margin. Hovy et al. (2011) showed how the sequential nature of the problem can be exploited in unsupervised learning. They present various sequential models and training options. They compare a standard bigram HMM and a very complex model that is designed to capture mutual constraints. In contrast to them, we use a trigram HMM, but move the preposition at the end of the observed sequence, to condition it on the previous words. As suggested there, we use EM with smoothing and random restarts.

2.3 Unsupervised POS-tagging

Merialdo (1994) introduced the task of unsupervised POS tagging using a dictionary. For each word, we know the possible labels in general. The model has to learn the labels in context. Subsequent work (Johnson, 2007; Ravi and Knight, 2009; Vaswani et al., 2010) has expanded on this in various ways, with accuracy between 86% and 96%. In this paper, we do not attempt to beat the state of the art, but rather test whether our constraints can be applied to a different task and data set.

3 Methodology

3.1 Data

For PSD, we use the SemEval task data. It consists of a training (16k) and a test set (8k) of sentences with sense-annotated prepositions following the sense inventory of *The Preposition Project*, TPP (Litkowski, 2005). It defines senses for each of the 34 most frequent English prepositions. There are on average 9.76 senses per preposition (between 2 and 25). We combine training and test and use the annotations from the training data to partially label our corpus. The test data remains unlabeled. We use the WordNet lexicographer senses as labels for the arguments. It has 45 labels for nouns, verbs, and adjectives and is thus roughly comparable to the prepositions sense granularity. It also allows us to construct a dictionary for the arguments from WordNet. Unknown words are assumed to have all possible senses applicable to their respective word class (i.e. all noun senses for words labeled as nouns, etc). We assume that pronouns other than "it" refer to people.

For the POS-tagged data, we use the Brown corpus. It contains $57k$ sentences and about $1,16m$ words. We assume a simplified tag set with 38 tags and a dictionary that lists all possible tags for each word. For the partial annotations, we label every occurrence of "the", "a", and "an" as DET, and the next word with possible tag NN as NN. Additional constraints label all prepositions as "P" and all forms of "be" as "V". We train on the top two thirds and test on the last third.

For both data sets, we converted all words to lower case and replaced numbers by "@".

3.2 Models

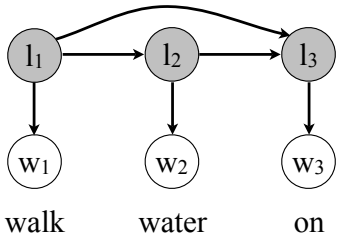

Figure 2: PSD: Trigram HMM with preposition as last element

For POS-tagging, we use a standard bigram HMM without back-off.

For PSD, we use a trigram HMM, but move the preposition at the end of the observed sequence, to condition it on the previous words (see Figure 2). Since not all prepositions have the same set of labels, we train individual models for each preposition. We can thus learn different parameter settings for the different prepositions.

We use EM with smoothing and random restarts to train our models. For smoothing, ϵ is added to each fractional count before normalization at each iteration to prevent overfitting (Eisner, 2002a). We

set ϵ to 0.01. We stop training after 40 iterations, or if the perplexity change between iterations was less than 0.0001. We experimented with different numbers of random restarts (none, 10, 50, and 100).

3.3 Dealing with Partial Annotations

The most direct way to constrain a specific word to only one label is to substitute it for a special token that has only that label. If we have a partially annotated example "walk on-$sense_5$ water" as input (see Figure 1), we add an emission probability $P(word = label|tag = label)$ to our model.

However, this is problematic in two ways. Firstly, we have effectively removed a great number of instances where "on" should be labeled "$sense_5$" from our training data, and replaced them with another token: there are now fewer instances from which we collect $C(on|sense_5)$. The fractional counts for our transition parameters are not affected by this, but the counts for emission parameter are skewed. We thus essentially siphon probability mass from $P(on|sense_5)$ and move it to $P(on : sense_5|sense_5)$. Since the test data never contains labels such as $sense_5$, our partial annotations have moved a large amount of probability mass to a useless parameter: we are never going to use $P(on : sense_5|sense_5)$ during inference!

Secondly, since EM tends to find uniform distributions (Johnson, 2007), other, rarer labels will also have to receive some probability. The counts for labels with partial annotations are fixed, so in order to use the rare labels (for which we have no partial annotations), their emission counts need to come from unlabeled instances. Say $sense_1$ is a label for which we have no partial annotations. Every time EM collects emission counts from a word "on" (and not a labeled version "on:$sense_n$"), it assigns some of it to $P(on|sense_1)$. Effectively, we thus assign too much probability mass to the emission of the word from rare labels.

The result of these two effects is the inverse of what we want: our model will use the label with the *least* partial annotations (i.e., a rare label) disproportionately often during inference, while the labels for which we had partial annotations are rarely used. The resulting annotation has a low accuracy. We show an example of this in Section 5.

The solution to this problem is simple: *parameter tying*. We essentially have to link each partial annotation to the original word that we replaced. The observed word "on" and the partial annotation "$on : sense_5$" should behave the same way during training. This way, our emission probabilities for the word "on" given a label (say, "$sense_5$") take the information from the partial annotations into account. This technique is also described in Eisner (2002b) for a phonological problem with similar properties. Technically, the fractional counts we collect for $C(on : sense_5|sense_5)$ should also count for $C(on|sense_5)$. By tying the two parameters together, we achieve exactly that. This way, we can prevent probability mass from being siphoned away from the emission probability of the word, and an undue amount of probability mass from being assigned to rare labels.

4 Experiments

4.1 How Much Annotation Is Needed?

In order to test the effect of partial annotations on accuracy, we built different training sets. We varied the amount of partial annotations from 0 to 65% in increments of 5%. The original corpus we use contains 67% partial annotations, so we were unable to go beyond this number. We created the different corpora by randomly removing the existing annotations from our corpus. Since this is done stochastically, we ran 5 trials for each batch and averaged the results.

We also test the effect more unsupervised data has on the task. Theoretically, unsupervised methods should be able to exploit additional training data. We use 27k examples extracted from the prepositional attachment corpus from Ratnaparkhi et al. (1994).

4.2 What Kind of Annotation Is Needed?

We can assume that not only the quantity, but also the distribution of the partial annotations makes a difference. Given that we can only annotate a certain percentage of the data, how should we best distribute those annotations among instances to maximize accuracy? In order to test this, we hold the amount of annotated data fixed, but vary the labels we use. We choose one sense and annotate only the instances that have that sense, while leaving the rest unlabeled.

Ideally, one would like to examine all subsets of annotations, from just a single annotation to all but one instances of the entire training data. This would cover the spectrum from unsupervised to supervised. It is unlikely that there is a uniform best number that holds for all problems within this immense search space. Rather, we explore two very natural cases, and compare them to the unsupervised case, for various numbers of random restarts:

1. all partial annotations are of the same sense
2. one labeled example of each sense

5 Results

System	Acc. (%)
semi-supervised w/o param tying	4.73
MFS baseline	40.00
unsupervised (Hovy et al., 2011)	55.00
semi-supervised, no RR	63.18
semi-supervised, 10 RR	63.12
semi-supervised, 50 RR	63.16
semi-supervised, 100 RR	63.22
semi-supervised, addtl. data, no RR	62.67
semi-supervised, addtl. data, 10 RR	62.47
semi-supervised, addtl. data, 50 RR	62.58
semi-supervised, addtl. data, 100 RR	62.58
supervised (Hovy et al., 2010)	84.50

Table 1: Accuracy of various PSD systems. Baseline is most frequent sense.

Table 1 shows the results for the PSD systems we tested. Since not all test sets are the same size, we report the weighted average over all prepositions. For significance tests, we use two-tailed t-tests over the difference in accuracy at $p < 0.001$.

The difference between our models and the baseline as well as the best unsupervised models in Hovy et al. (2011) are significant. The low accuracy achieved without parameter tying underscores the importance of this technique. We find that the differences between none and 100 random restarts are not significant if partial annotations are used. Presumably, the partial annotations provide a strong enough constraint to overcome the effect of the random initializations. I.e., the fractional counts from the partial annotations overwhelm any initial parameter settings and move the model to a more advantageous position in the state space. The good accuracy for the case with no restarts corroborates this.

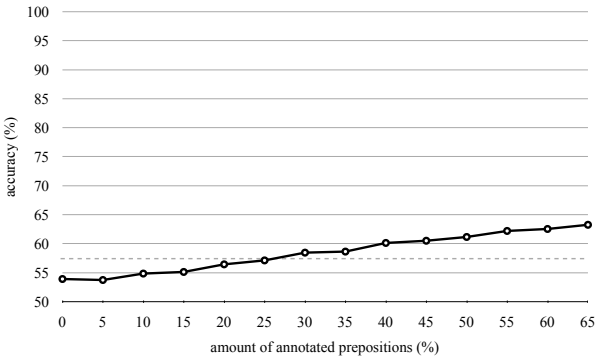

Figure 3: Accuracy for PSD systems improves linearly with amount of partial annotations. Accuracies above dotted line improve significantly (at $p < 0.001$) over unsupervised approach (Hovy et al., 2011)

Figure 3 shows the effect of more partial annotations on PSD accuracy. Using no annotations at all, just the dictionary, we achieve roughly the same results as reported in Hovy et al. (2011). Each increment of partial annotations increases accuracy. At around 27% annotated training examples, the difference starts to be significant. This shows that unsupervised training methods can benefit from partial annotations. When adding more unsupervised data, we do not see an increase in accuracy. In this case, the algorithm failed to make use of the additional training data. This might be because the two data sets were not heterogenous enough, or because the number of emission parameters grew faster than the amount of available training examples. A possible, yet somewhat unsatisfying explanation is that when we increase the overall training data, we reduce the percentage of labeled data (here to 47%; the result was comparable to the one observed in our ablation studies). It seems surprising, though, that the model does not benefit from the additional data[1]. More aggressive smoothing might help alleviate that problem.

The results on the distribution of partial annotation are shown in Figure 4. Using only the most

[1]Note that similar effects were observed by (Smith and Eisner, 2005; Goldwater and Griffiths, 2007).

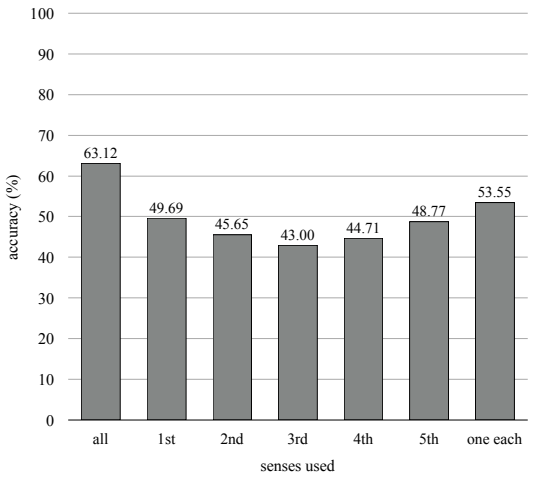

Figure 4: Labeling one example of each sense yields better results than all examples of any one sense. Senses ordered by frequency

frequent sense, accuracy drops to 49.69%. While this is better than the baseline which simply assigns this sense to every instance, it is a steep drop. We get better results using just one annotated example of each sense (53.55%).

System	Acc. (%)
(Merialdo, 1994)	86.60
random baseline	62.46
unsupervised, no RR	82.77
semi-supervised, DET+NN	88.51
semi-supervised, DET+NN+P	88.97
semi-supervised, DET+NN+P+V	87.07

Table 2: Accuracy of various POS systems. Random baseline averaged over 10 runs.

The results for POS tagging confirm our previous findings. The random baseline chooses for each word one of the possible tags. We averaged the results over 10 runs. The difference in accuracy between both the baseline and the unsupervised approach as well as the unsupervised approach and any of the partial annotations are significant. However, the drop in accuracy when adding the last heuristic points to a risk: partial annotation with heuristics can introduce errors and offset the benefits of the constraints. Careful selection of the right heuristics and the tradeoff between false positives they introduce and true positives they capture can alleviate this problem.

6 Related Research

Unsupervised methods have great appeal for resource-poor languages and new tasks. They have been applied to a wide variety of sequential labeling tasks, such as POS tagging, NE recognition, etc. The most common training technique is forward-backward EM. While EM is guaranteed to improve the data likelihood, it can get stuck in local maxima. Merialdo (1994) showed how the the initialized model influences the outcome after a fixed number of iterations. The importance is underscored succinctly by Goldberg et al. (2008). They experiment with various constraints.

The idea of using partial annotations has been explored in various settings. Druck et al. (2008) present an approach to label features instead of instances for discriminative probabilistic models, yielding substantial improvements. They also study the effectiveness of labeling features vs. labeling instances. Rehbein et al. (2009) study the utility of partial annotations as precursor to further, human annotation. Their experiments do not extend to unsupervised training. Tsuboi et al. (2008) used data that was not full annotated. However, their setting is in principle supervised, only few words are missing. Instead of no labels, those words have a limited number of possible alternatives. This works well for tasks with a small label alphabet or data where annotators left multiple options for some words. In contrast, we start out with unannotated data and assume that some words can be labeled. Gao et al. (2010) present a successful word alignment approach that uses partial annotations. These are derived from human annotation or heuristics. Their method improves BLEU, but requires some modification of the EM framework.

7 Conclusion and Future Work

It is obvious, and common knowledge, that providing some annotation to an unsupervised algorithm will improve accuracy and learning speed. Surprisingly, however, our literature search did not turn up any papers stating exactly how and to what degree the improvements appear. We therefore selected a

very general training method, EM, and a simple approach to include partial annotations in it using parameter tying. This allows us to find more stable starting points for sequential labeling tasks than random or uniform initialization. We find that we would need a substantial amount of additional unlabeled data in order to boost accuracy. In contrast, we can get significant improvements by partially annotating some instances (around 27%). Given that we can only annotate a certain percentage of the data, it is best to distribute those annotations among all applicable senses, rather than focus on one. This obviates the need for random restarts and speeds up training.

This work suggests several interesting new avenues to explore. Can one integrate this procedure into a large-scale human annotation effort to obtain a kind of active learning, suggesting which instances to annotate next, until appropriate stopping criteria are satisfied (Zhu et al., 2008)? Can one determine upper bounds for the number of random restarts given the amount of annotations?

Acknowledgements

We would like to thank Victoria Fossum, Kevin Knight, Zornitsa Kozareva, and Ashish Vaswani for invaluable discussions and advice. We would also like to thank the reviewers who provided us with helpful feedback and suggestions. Research supported in part by Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program.

References

Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 280–287, Prague, Czech Republic. Association for Computational Linguistics.

Ming-Wei Chang, Lev Ratinov, Nicholas Rizzolo, and Dan Roth. 2008. Learning and inference with constraints. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 1513–1518.

Gregory Druck, Gideon Mann, and Andrew McCallum. 2008. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602. ACM.

Jason Eisner. 2002a. An interactive spreadsheet for teaching the forward-backward algorithm. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, pages 10–18. Association for Computational Linguistics.

Jason Eisner. 2002b. Parameter estimation for probabilistic finite-state transducers. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 1–8. Association for Computational Linguistics.

Kuzman Ganchev, João Graça, Jennifer Gillenwater, and Ben Taskar. 2010. Posterior regularization for structured latent variable models. *The Journal of Machine Learning Research*, 11:2001–2049.

Qin Gao, Nguyen Bach, and Stephan Vogel. 2010. A semi-supervised word alignment algorithm with partial manual alignments. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR*, pages 1–10. Association for Computational Linguistics.

Yoav Goldberg, Meni Adler, and Michael Elhadad. 2008. Em can find pretty good hmm pos-taggers (when given a good start). In *Proceedings of ACL*.

Sharon Goldwater and Thomas Griffiths. 2007. A fully bayesian approach to unsupervised part-of-speech tagging. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 744.

João Graça, Kuzman Ganchev, and Ben Taskar. 2007. Expectation maximization and posterior constraints. *Advances in Neural Information Processing Systems*, 20:569–576.

Dirk Hovy, Stephen Tratz, and Eduard Hovy. 2010. What's in a Preposition? Dimensions of Sense Disambiguation for an Interesting Word Class. In *Coling 2010: Posters*, pages 454–462, Beijing, China, August. Coling 2010 Organizing Committee.

Dirk Hovy, Ashish Vaswani, Stephen Tratz, David Chiang, and Eduard Hovy. 2011. Models and training for unsupervised preposition sense disambiguation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 323–328, Portland, Oregon, USA, June. Association for Computational Linguistics.

Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 296–305.

Ken Litkowski and Orin Hargraves. 2007. SemEval-2007 Task 06: Word-Sense Disambiguation of Prepositions. In *Proceedings of the 4th International*

Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic.

Ken Litkowski. 2005. The preposition project. http://www.clres.com/prepositions.html.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational linguistics*, 20(2):155–171.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Proceedings of the workshop on Human Language Technology*, pages 250–255. Association for Computational Linguistics.

Sujith Ravi and Kevin Knight. 2009. Minimized models for unsupervised part-of-speech tagging. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 504–512. Association for Computational Linguistics.

Ines Rehbein, Josef Ruppenhofer, and Caroline Sporleder. 2009. Assessing the benefits of partial automatic pre-labeling for frame-semantic annotation. In *Proceedings of the Third Linguistic Annotation Workshop*, pages 19–26. Association for Computational Linguistics.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Stephen Tratz and Dirk Hovy. 2009. Disambiguation of Preposition Sense Using Linguistically Motivated Features. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 96–100, Boulder, Colorado, June. Association for Computational Linguistics.

Yuta Tsuboi, Hisashi Kashima, Hiroki Oda, Shinsuke Mori, and Yuji Matsumoto. 2008. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics*, volume 1, pages 897–904. Association for Computational Linguistics.

Ashish Vaswani, Adam Pauls, and David Chiang. 2010. Efficient optimization of an mdl-inspired objective function for unsupervised part-of-speech tagging. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 209–214. Association for Computational Linguistics.

Jingbo Zhu, Huizhen Wang, and Eduard Hovy. 2008. Multi-criteria-based strategy to stop active learning for data annotation. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 1129–1136. Association for Computational Linguistics.