Unsupervised Bilingual POS Tagging with Markov Random Fields

Desai Chen
desaic@andrew.cmu.edu

advisor: Noah A. Smith
nasmith@cs.cmu.edu

Abstract

This paper explores unsupervised learning with undirected graphical models. We focus on the problem of bilingual part of speech (POS) induction, which considers the POS induction problem when parallel training data is available [Snyder et al., 2008]. Because we use undirected models, there are no restrictions on the structure of the graphs and we can incorporate many overlapping features, such as sublexical features (this is in contrast to previous work which made use of directed, generative models). Although our undirected model is quite flexible in terms of being able to add new features, the unsupervised learning problem turns out to be quite challenging, and analysis determines that the non-convex objective we are attempting to optimize has many local optima which causes problems for learning. We show that performance can be improved by using an alternative objective based on contrastive estimation [Smith and Eisner, 2005b].

Contents

1 Introduction

Unsupervised learning has received considerable attention in natural language processing. It has the advantage of compensating for missing annotations, and can also provide computational models for language understanding and learning.

While unsupervised learning holds much promise, the performance of unsupervised systems lags significantly behind supervised systems. This is true with various basic NLP tasks such as part-of-speech tagging and parsing. Recently, the effectiveness of *multilingual* learning has been demonstrated for various tasks [Snyder et al., 2008, Cohen and Smith, 2009, Berg-Kirkpatrick et al., 2010], and has shown to narrow the gap between supervised systems and unsupervised systems.

In this paper, we look at how model parameterization affects the quality of the learned model. We are motivated by the observation that in supervised learning, globally normalized undirected models are widely used. These models provide a principled way to incorporate arbitrary, possibly correlated features in the model. This is attractive because feature engineering can be used to include knowledge in the model. Conditional random fields [Lafferty et al., 2001] are conditionally trained undirected graphical models that are widely used. In general, these undirected models have been found to perform better than directed, locally normalized models [Toutanova et al., 2003]. While conditional discriminative undirected models are prevailing, joint MRF models are far less widely used in NLP. In this paper, we explore the hypothesis that undirected models will also improve unsupervised learning, because feature engineering can be used to encode useful knowledge about the problem space.

In this paper, we consider the problem of unsupervised part-of-speech tagging in a multilingual setting. Like previous authors [Snyder et al., 2008, Das and Petrov, 2011], we assume that multilingual *parallel* data is available. The linguistic insight should be obvious. Having more languages provides additional information. Words of similar grammatical functions tend to be aligned together. This phenomenon is very strong in the training data.

Unlike previous approaches with a directed model, we propose an undirected model that can incorporate correlated features. In this paper we used prefix and suffix of words as additional features. Another advantage is that undirected models don't have to be acyclic.

Yet, there are significant challenges when using undirected models. For example, one problem is that MRFs are hard to train. Exact inference is almost impossible because our graph contains cycles. Training such MRFs requires approximate inference techniques. The behavior of a combination of unsupervised MRFs and approximate inference is an interesting research question. In contrast

to MRFs, directed models have a set of well-established techniques to train [Goldwater and Griffiths, 2007, Toutanova and Johnson, 2007, Johnson, 2007, Snyder et al., 2009]. Koller and Friedman [2009] provides a comprehensive comparison of directed Bayesian generative graphical models and undirected graphical models. In this paper we provide a comprehensive comparison of the difficulty of training between directed and undirected models.

2 Related work

2.1 Bayesian Directed Model

The model most relevant to our work is the one presented by Snyder et al. [2008]. The part of speech induction model can be represented as a graphical model as in Figure 1. The structure looks like two Hidden Markov Models (HMMs).

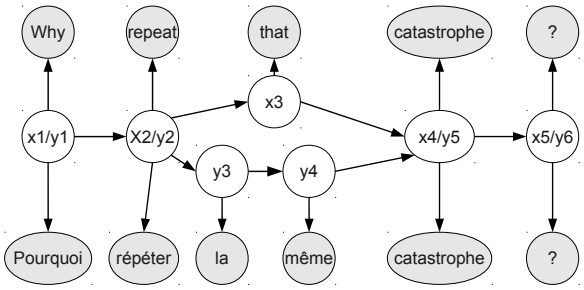

Figure 1: Bilingual Directed POS induction model

More formally, suppose we have two sentences $\mathbf{s} = (s_1, s_2, ..., s_{N_s})$ and $\mathbf{t} = (t_1, t_2, ..., t_{N_t})$ that translates to each other. Call one of them the source sentence and the other the target sentence. N_s and N_t are the lengths of the source sentence and the target sentence. Let $\mathbf{x} = (x_1, x_2, ..., x_{N_s}), \mathbf{y} = (y_1, y_2, ..., y_{N_t})$ be a tagging of the sentence pair. x_i is the POS tag for the ith word in the source sentence and y_i is the POS tag for the ith word in the target sentence. Define A as the set of word alignment links such that $(i, j) \in A$ if and only if $\mathbf{s}_i$ is aligned to $\mathbf{t}_j$. Given a monotonic alignment A (alignment with no crossing edges), the graphical model is constructed by merging the tags of aligned word pairs. In practice, it's very common to have word alignment with *crossing* links. Since in a directed model, cycles are not allowed, these crossing links have to be removed by some kind of heuristic. For example, one can go through the alignment links in the order of source side sentence and then remove any links that intersect with preceding links. The tags are merged instead of being separate variables because all the edges have to

be directed and it is not clear how to define directions between aligned tags. The values in such merged nodes are, therefore, pairs of tags (x_i, y_j). This model is called a merged tag model, named from the merging operation. This is a directed model, each random variable is generated from a locally normalized distribution conditioned on its parents. The tag sequence is generated from left to right and the words are generated by the tags as the arrows indicates. The joint probability of two sentences and their tagging is

$$P(x_1, ..., x_{N_s}, y_1, ..., y_{N_t}, \mathbf{s}, \mathbf{t}) = \prod_{(i,j)\in A} P(x_i, y_j | x_{i-1}, y_{j-1}) P(\mathbf{s}_i | x_i) P(\mathbf{t}_j | y_j) \cdot$$

$$\prod_{\text{unaligned } i} P(x_i | x_{i-1}) P(\mathbf{s}_i | x_i) \cdot \prod_{\text{unaligned } j} P(y_j | y_{j-1}) P(\mathbf{t}_j | y_j).$$

Merged nodes represent the distribution of two tags of aligned words using additional coupling parameters $\omega(x_i, y_j)$. The transition probability is given by

$$P(x_i, y_j | x_{i-1}, y_{j-1}, \omega) = \frac{1}{Z}(P(x_i | x_{i-1}) P(y_i | y_{i-1}) \omega(x_i, y_j)).$$

Z is the normalization coefficient computed from all combinations of x_i and y_j.

$$Z = \sum_{x,y} P(x | x_{i-1}) P(y | y_{i-1}) \omega(x, y)$$

In order to guide the unsupervised learning process, all the transition, emission and coupling parameters are governed by Dirichlet priors as commonly done with Bayesian models. The inference procedure is a Gibbs sampling based approach. To sample a word, we compute the emission probability of a word s_i given all other words $\mathbf{s}_{\setminus i}$ in the training data, its tag x_i and Dirichlet prior θ

$$P(s_i | x_i, \mathbf{s}_{\setminus i}, \theta_0) = \frac{n(x_i, s_i) + \theta_0}{n(x_i) + W_{x_i}\theta_0}.$$

$n(x_i, s_i)$ is number of times tag x_i co-occur with word s_i, $n(x_i)$ is the number of times x_i appears in the data, W_{x_i} is the total number of word types that can be emitted from x_i. This can be seen as estimating the conditional distribution with add θ_0 smoothing. To sample a tag, a set of transition probabilities are sampled based on the prior, and then a tag is sampled using the transition probabilities. The hyper-parameters of the prior distributions are also inferred from data.

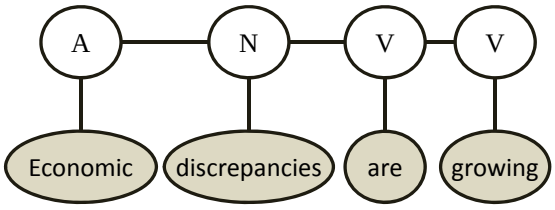

Figure 2: Monolingual tag model.

2.2 Undirected Model for POS labelling

Undirected models look very much like directed models. The only difference in terms of the graphical representation of these models is that the edges do not have directionality. The small difference in definition leads to very different kind of representation of probability distributions.

When the tags are observed in the supervised setting, Conditional random fields [Lafferty et al., 2001] outperform directed models because they can incorporate features.

In the unsupervised setting, Smith and Eisner [2005b] shows that a monolingual part of speech tag induction model can be improved with spelling features (prefixes and suffixes of words). Haghighi and Klein [2006] shows that distributional similarity features can also improve the performance. An example of such a monolingual model is shown in Figure 2.

In the monolingual case, suppose we have a sentence $\mathbf{s}$ and a tagging $\mathbf{x}$. For simplicity of notation, we are only writing out formulas for one sentence. The expression for the likelihood and gradient can be easily generalized to multiple sentences by adding the same expression for each sentences. $\mathbf{w}$ are parameters chosen to maximize the data likelihood

$$\mathcal{L}(\mathbf{w}) = \log P(\mathbf{s}|\mathbf{w}) = \log \sum_{\mathbf{x}} P(\mathbf{s}, \mathbf{x}|\mathbf{w}).$$

The joint probability of a sentence and a particular tagging $\mathbf{x}$ is

$$P(\mathbf{s}, \mathbf{x}|\mathbf{w}) = \frac{1}{Z(\mathbf{w})} \prod_{i=1}^{N} \exp(\mathbf{w} \cdot (\mathbf{f}(s_i, x_i) + \mathbf{f}(x_i, x_{i+1})).$$

$\mathbf{f}(s_i, x_i)$ is a feature vector defined over edges between a tag x_i and word s_i. $\mathbf{f}(x_i, x_{i+1})$ is a feature vector defined over edges between adjacent tags. $Z(\mathbf{w})$

is the partition function over possible word sequences

$$
\begin{aligned}
Z(\mathbf{w}) &= \sum_{\mathbf{s}} \sum_{\mathbf{x}} \prod_{i=1}^{N} \exp(\mathbf{w} \cdot (\mathbf{f}(s_i, x_i) + \mathbf{f}(x_i, x_{i+1})) \\
&= \sum_{\mathbf{s}} \sum_{\mathbf{x}} \text{score}(\mathbf{s}, \mathbf{x}).
\end{aligned} \tag{1}
$$

Undirected models are usually trained with gradient based methods both in supervised and unsupervised settings. The partial derivative of the likelihood function with a particular weight w_i can be derived to take the following form

$$
\frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_i} = (E_{\mathbf{x}|\mathbf{s},\mathbf{w}}[f_i] - E_{\mathbf{x},\mathbf{s}|\mathbf{w}}[f_i]).
$$

The first term is the expectation of features under the distribution $P(\mathbf{x}|\mathbf{s}, \mathbf{w})$. It can be computed with the same forward-backward algorithm used for CRF. The second term is harder. Fortunately, it does not depend on the data at all, so that it only needs to be computed once per iteration. Haghighi and Klein [2006] approximated the second term by ignoring all sentences of length greater than a constant L.

$$
E_{\mathbf{x},\mathbf{s}|\mathbf{w}}[f_i] = \sum_{l=1}^{L} E_{\mathbf{x},\mathbf{s}|l,\mathbf{w}}[f_i].
$$

Each term can be computed by a slight modification of the forward-backward algorithm. This is a reasonable approximation because very long sentences are very unlikely to appear in data. Another reason for the approximation is from computation concerns. It is very hard to compute the expected value of features for all possible sentences with length ranging from 0 to infinity. Smith and Eisner [2005b] provides many other ways to approximate the second term of the gradient. The general method for approximating the second term is named contrastive estimation. It is an indispensable component of our training algorithm for our new model.

3 Model

Our model is based on a Markov random field which consists of observed lexical nodes for two languages and latent nodes representing the part-of-speech tags for the lexical nodes. An example of this Markov random field for a pair of sentences in French and English is given in Figure 3 in graphical model notation.

More formally, suppose we have two sentences $\mathbf{s} = (s_1, s_2, ..., s_{N_s})$, and $\mathbf{t} = (t_1, t_2, ..., t_{N_t})$, with corresponding tagging sequences $\mathbf{x} = (x_1, x_2, ..., x_{N_s})$ and

$\mathbf{y} = (y_1, y_2, ..., y_{N_t})$, and word alignment A as defined in Section 2.1. The joint distribution of words and tags is given by

$$
\begin{aligned}
& p(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t} \mid A, \mathbf{w}) \propto \\
& \prod_{i=1}^{N_s} \exp(\mathbf{w} \cdot \mathbf{f}(s_i, x_i)) \times \prod_{i=1}^{N_t} \exp(\mathbf{w} \cdot \mathbf{f}(t_i, y_i)) \\
& \prod_{i=1}^{N_s - 1} \exp(\mathbf{w} \cdot \mathbf{f}(x_i, x_{i+1})) \times \prod_{i=1}^{N_t - 1} \exp(\mathbf{w} \cdot \mathbf{f}(y_i, y_{i+1})) \\
& \prod_{(i,j) \in A} \exp(\mathbf{w} \cdot \mathbf{f}(x_i, y_j))
\end{aligned}
\tag{2}
$$

A lower case "p" denotes an unnormalized score of the random variables. An upper case "P" would denote a proper probability in the following formulas. Each term of the form $\exp(\mathbf{w} \cdot \mathbf{f}(\bullet, \bullet))$ is called a factor. In theory one can define an arbitrary way of computing the factors for each edge. Exponentiating the linear combination of feature values is preferred because the gradient of the weights with respect to the marginal probability of the observed words work out to have very simple form. The marginal probability of $\mathbf{s}, \mathbf{t}$ is given by

$$
P(\mathbf{s}, \mathbf{t}) = \frac{\sum_{\mathbf{x},\mathbf{y}} p(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t})}{\sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} p(\mathbf{s}', \mathbf{t}', \mathbf{x}, \mathbf{y})}. \tag{3}
$$

The gradient of the log of the probability with respect to a weight w_j is then

$$
\frac{\partial \log P(\mathbf{s}, \mathbf{t}_i)}{\partial w_j} = \mathbf{E}[f_j | \mathbf{s}, \mathbf{t}] - \mathbf{E}[f_j]. \tag{4}
$$

The gradient for w_j is the expected value of feature f_j conditioned on the observed sentence pair minus the expected value of f_j over all possible configuration of random variables.

The features we used in this model include conjunctions of a tag and a word, conjunctions of adjacent tags, and conjunctions of a tag and another tag if the edge represents an alignment link. These features model the information the Bayesian HMM from Section 2.1 captures. In addition, we used conjunctions of prefixes and suffixes of a word and its tag. The full details about these features are given in Table 1. We also experimented with other features such as character trigrams and indicators of numbers but the results are not much different. If we have more time in the future , we would first try distributional similarity of words.

Using a Markov random field, as opposed to directed models has the advantage of being able to incorporate alignments with *crossing* links. The intuitive reason

Feature Name		Description
Emission Feature	word	$f_{s,x}(s', x') = 1$ if $s' = s$ and $x' = x$.
	prefix	$f_{x,p(s)}(x', s') = 1$ if $x' = x$ and $p(s') = p(s)$
Transition Feature		$f_{x_1,x_2}(x'_1, x'_2) = 1$ if $x'_1 = x_1$ and $x'_2 = x_2$
Alignment Feature		$f_{x,y}(x', y') = 1$ if $x' = x$ and $y' = y$

Table 1: Features used for the model. x or y denotes a tag in the source or target language, s or t denotes a word, $p(s)$ denotes a prefix or suffix of a word

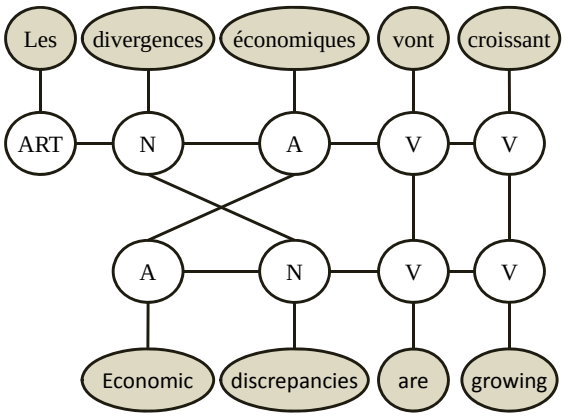

Figure 3: Bilingual tag model.

is that in a directed model, an edge denotes a causal relation, if there is a loop in the graph, it means something is causing itself. In an MRF, each edge denotes some kind of relation, and therefore an MRF doesn't care about creating cycles. For example, Figure 3 shows a crossing link between "Economic discrepancies" and "divergences economiques." This crossing link is indicative about the relation between "Economic" and "economiques." We will show that crossing links make a difference in the experiments section.

4 Inference and Parameter Learning

Similar to the unsupervised monolingual MRF model in Section 2.2, we find the features weights $\mathbf{w}$ to maximize the log-likelihood with gradient descent. The difference between our model and previous monolingual model lies in the graphical structure. In the bilingual case, the graph is no longer a linear chain. The graph can contain cycles. Dynamic programming algorithms for computing the gradi-

ent exactly are no longer applicable. We developed a new sampling scheme for computing the gradient. We also found that the likelihood objective contains many local optima that correspond to bad tagging. Contrastive estimation turns out to be a very handy technique to guide the model by simplifying the objective function. Computationally, contrastive estimation fits nicely within the sampling scheme.

4.1 MLE with Gradient Descent

Following the notations in Equation 3 in Section 3, the log-likelihood of the data given $\mathbf{w}$ is

$$\begin{aligned}\mathcal{L}(\mathbf{w}) &= \log P(\mathbf{s},\mathbf{t}) \\ &= \log \sum_{\mathbf{x},\mathbf{y}} p(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}) - \log \sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} p(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}) \\ &= \log \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t})) - \log \sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y})).\end{aligned} \tag{5}$$

Using chain rule of derivatives, we can derive the partial derivative of the log-likelihood with respect to a weight w_i.

$$\begin{aligned}\frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_i} &= \frac{\partial \log P(\mathbf{s},\mathbf{t})}{\partial w_i} \\ &= \frac{\partial \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}))}{\sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}))} - \frac{\partial \sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}))}{\sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}))} \\ &= \frac{\partial \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}))}{Z_{\mathbf{s},\mathbf{t}}(\mathbf{w})} - \frac{\partial \sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} \exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}))}{Z(\mathbf{w})} \\ &= \sum_{\mathbf{x},\mathbf{y}} f_i(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}) \frac{\exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}))}{Z_{\mathbf{s},\mathbf{t}}(\mathbf{w})} - \\ &\sum_{\mathbf{s}',\mathbf{t}'} \sum_{\mathbf{x},\mathbf{y}} f_i(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}) \frac{\exp(\mathbf{w}\cdot\mathbf{f}(\mathbf{s}',\mathbf{t}',\mathbf{x},\mathbf{y}))}{Z(\mathbf{w})} \\ &= \mathbf{E}_{P(\mathbf{x},\mathbf{y}|\mathbf{s},\mathbf{t})}[f_i] - \mathbf{E}_{P(\mathbf{s},\mathbf{t})}[f_j].\end{aligned} \tag{6}$$

These two terms can both be approximated with Gibbs sampling. A general explanation of Gibbs sampling technique is in Section 4.2. In the bilingual case, since the graphical model is not a linear chain and contains loops, the forward-backward algorithm cannot be used to compute expectation. This situation is different from previous works on linear chain structured MRF [Smith and Eisner, 2005b, Haghighi and Klein, 2006] which have used exact inference algorithm to compute the gradient.

The first term of the gradient in Equation 4 is the expected values of feature functions conditioned on words. This can be computed by keeping the words fixed and sampling tags many times. As explained in Section 4.2, Gibbs sampling works by picking each random conditioned on all other random variables as in Equation 7. In an undirected graphical model, a random variable only depend on its direct neighbors when all other variables are observed. The distribution of a tag x_i is determined by the corresponding word s_i, adjacent tags x_{i+1}, x_{i-1} and aligned tag y_j if there is one. This means that in Equation 7, $p(x_i, \mathbf{x}_{\setminus i}) = p(x_i, x_{i-1}, x_{i+1}, s_t, y_j)$. A diagram of the Markov blanket of a tag is shown in Figure 4.

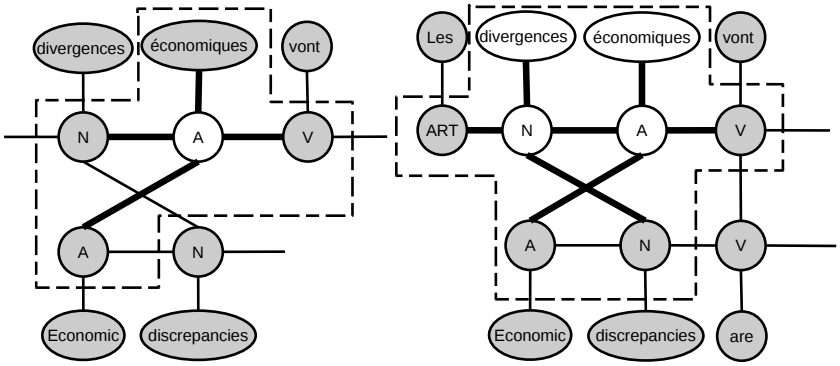

Figure 4: Markov blankets of a tag (left) and a pair of tags and words(right). Shaded nodes are fixed while sampling unshaded nodes. Thickened edges are relevant edges in determining the distribution. To sample a tag, a random tag is picked from the distribution of tags conditioned on its neighbours in the left figure. The probability of swapping a pair of tags and words simultaneously is computed based on the thickened edges in the right figure.

The second term of the gradient in Equation 4 is the expected value averaged over all possible word sequences. This expectation is harder to compute than the same expectations in linear chain models. In linear chain models, this term does not depend on the data and only needs to be computed once per iteration. In the bilingual case, the second term needs to sum over all possible alignment structures. Since computing for each alignment structure is very time consuming, we decided to compute this term for each sentence pair conditioned on its alignment structure. For a Gibbs sampler, this only means that it is sampling from a much larger space, while the complexity of sampling a new variable is asymptotically the same. It is hard to quantify how many more examples we need now that the space is so much larger.

In practice, we found that maximizing the likelihood didn't work well. We used

the idea of contrastive estimation and limited the sampler for the second term to only sample permutations of the words in the sentence. Details of using contrastive estimation is explained in Section 4.3.

4.2 Gibbs Sampling

Gibbs sampling is a major component of the training algorithm for our model. It is a very simple Markov Chain Monte Carlo method and can be seen as a special case of the Metropolis-Hastings algorithm [Bishop, 2006]. Its correctness can be shown based on the correctness of the Metropolis-Hastings algorithm. The Metropolis-Hastings algorithm obtains a sequence of samples from a complicated distribution when direct sampling is hard. It can be shown that the sequence of samples obtained approximates the original distribution. The proof that Gibbs sampling samples from the joint distribution of the random variables follows from the correctness of the Metropolis-Hastings algorithm.

Gibbs sampling samples a set of mutually dependent random variables by changing one variable at a time. Suppose we have a distribution $p(\mathbf{x})$ over N random variables. Each assignment of the random variable can be viewed as a state and Gibbs sampling moves between these states such that the probability of reaching at any state $\mathbf{x}$ is the same as the probability of that state defined by $p(\mathbf{x})$. Based on this intuition, a necessary requirement for Gibbs sampling to work is that all states should be reachable from other states.

Gibbs sampling starts at a random point $\mathbf{x}_0$. The sampler then loops through each random variable and picks a value for that random variable conditioned on all other random variables. This procedure is repeated many times to obtain a good estimate of probabilities. Suppose the sampler is at a state $\mathbf{x}$ and it wants to pick a new value for x_i. Denote $\mathbf{x}_{\setminus i}$ as the set of random variables excluding x_i. The sampler picks a value from the following distribution

$$p(x_i|\mathbf{x}_{\setminus i}) = \frac{p(x_i, \mathbf{x}_{\setminus i})}{\sum_{x_i'} p(x_i', \mathbf{x}_{\setminus i})}. \tag{7}$$

This conditional probability can be easily computed especially if x_i only depends on very few of the random variables. If we were to sample from the complete distribution instead, where each random can take k values, then we need to compute a distribution for N^k values. The main advantage of Gibbs sampling is that it provides an approximation to the joint distribution very efficiently. Each step only computes a distribution for k values. A sampling round through all random variables only needs to compute distribution for Nk values.

To address the requirement that all states should be reachable as mentioned in

the beginning of this section, it is sufficient to satisfy

$$p(x_i|\mathbf{x}_{\setminus i}) > 0, \forall \mathbf{x}.$$

Gibbs sampling can be viewed as a special case for Metropolis-Hastings algorithm in the following way. Suppose the sampler wants to move from state $\mathbf{x}$ to state $\mathbf{x}'$ where x_i is changed to x'_i. The remaining variables are the same, $\mathbf{x}_{\setminus i} = \mathbf{x}'_{\setminus i}$. Use the proposal probability $q(\mathbf{x}'|\mathbf{x}) = p(x'_i|\mathbf{x}_{\setminus i})$, the acceptance probability in Metropolis-Hastings algorithm can be shown to always equal to 1 using chain rule of probabilities.

$$\begin{aligned} a &= \frac{p(\mathbf{x}')q(\mathbf{x}|\mathbf{x}')}{p(\mathbf{x})q(\mathbf{x}'|\mathbf{x})} \\ &= \frac{p(x'_i|\mathbf{x}'_{\setminus i})p(\mathbf{x}'_{\setminus i})p(x_i|\mathbf{x}'_{\setminus i})}{p(x_i|\mathbf{x}_{\setminus i})p(\mathbf{x}_{\setminus i}))p(x'_i|\mathbf{x}_{\setminus i})} \\ &= 1 \end{aligned} \tag{8}$$

Using results from Metropolis-Hastings algorithm, we know that the accept rate of Gibbs sampling is 1. It always samples from the right distribution. However, it is not clear how many samples it takes to get a good estimation of quantities needed by our model.

4.3 Contrastive Estimation

We used a combination of Gibbs sampling [Casella and George, 1992] and contrastive estimation [Smith and Eisner, 2005a] to estimate the parameters. Contrastive estimation approximates the log-likelihood with an easier objective function. It limits the space of possible sentences to a much smaller subset $N(\mathbf{x})$ called the neighborhood of $\mathbf{x}$. The objective function is modified slightly to

$$\begin{aligned} \mathcal{L}(\mathbf{w}) &= \log P(\mathbf{s}, \mathbf{t}) \\ &= \log \sum_{\mathbf{x},\mathbf{y}} p(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t}) - \log \sum_{\mathbf{s}',\mathbf{t}' \in N(\mathbf{s}',\mathbf{t}')} \sum_{\mathbf{x},\mathbf{y}} p(\mathbf{s}', \mathbf{t}', \mathbf{x}, \mathbf{y}). \end{aligned} \tag{9}$$

Same as in Section 3, a lower case "p" denotes an unnormalized score of random variables. The likelihood is more like a supervised objective where the model needs to discriminate positive examples from negative examples. The gradient is then modified accordingly where the second term of the expectation only sums over sentences in the neighborhood.

In our model, we used the Dynasearch neighborhood [Potts and van de Velde, 1995, Congram et al., 2002]. It is a subset of permutations of sentence $\mathbf{x}$. A word

can only go to its adjacent positions or stay where it is. To make sure that sentences in the neighborhood are mostly negative examples, we rely on the linguistic observation that in a language, permutations of words are probably not well-formed sentences syntactically.

The training algorithm iterates between sampling part-of-speech tags and sampling permutations of words to compute the expected value of features. At each step, the sampler decides whether to swap a pair of adjacent tags and words or not. The Markov blanket for computing probability of swapping a pair of tags and words is shown in Figure 4.

By limiting the space of possible sentences, the model is driven to pay more attention to more specific features of a language. When the space is defined as neighborhood of permutations, the model will try to set the weights so that the sentences are more likely to keep in the original order. The model does so by putting more weights on transition features and word alignment features to focus on relations between words. In contrast, when the space is all word sequence pairs that can fit into the alignment structure, we observed that the model tends to pay more attention to emission features and focus on the distribution of the vocabulary.

5 Experiments

We tested our model on several language pairs mainly with two settings. The two settings differ in amount of tag dictionary we have. A tag dictionary contains a list of words and for each word it lists possible tags that word can take. For example, in an English tag dictionary, the word "fish" may contain two options, "N" for noun or "V" for verb. The presence of tag dictionary makes the unsupervised problem much less ambiguous and makes the inference procedure much faster. The first setting uses a complete tag dictionary for each word appeared in the training data just to verify that our model is reasonable. It is often very difficult to obtain a tag dictionary for all words that appeared in the data. In practice, it may be much easier to let a person with basic linguistic knowledge to write down a small tag dictionary for frequent words. The second setting uses a tag dictionary for top 100 most frequent words.

We chose two datasets to evaluate the tagging accuracy of our models. The first dataset, the Orwell novel *1984*, is chosen to compare to previous state-of-art bilingual POS tag induction. The second piece of data is a parallel corpus named EUROPARL (Koehn [2005]). It is a much bigger dataset so that we can see how our model scales in terms of speed and accuracy.

Tag	Part of speech
A	Adjectives
C	Conjunctions
D	Determiners
M	English word for numbers and arabic numbers
N	All nouns
P	Pronouns
PUN	Punctuations in the middle of a sentence
R	Adverbs
S	Prepositions
V	All verbs (action, modal, auxiliary)
X	Unknown
Y	Not used
.	Punctuations at the end of a sentence

Table 2: List of part of speech tags for English.

5.1 Training and test data

The first data set includes parallel text of the 1984 novel in English, Bulgarian, Slovene and Serbian [Erjavec, 2004]. The data set is manually annotated with part-of-speech tags. We use automatically induced words alignments from Giza++ [Och and Ney, 2003] following Snyder et al. [2008]. The data shows very regular patterns of tags that are aligned together: words with the same tag in two languages tend to be aligned with each other.

The 1984 data set includes fourteen part-of-speech tags, two of which denote punctuations. The tag set for English is listed in Table 2. Tag sets for other languages have minor differences in determiners and particles.

We ran our model on this dataset with both complete and partial tag dictionary. The tag dictionary for languages other than English are produced using the training data. When a complete tag dictionary is present, each word has only a small number of tags it can possibly use. The baseline of choosing random tags for each word gives an accuracy of around 85%. English has an extended tag dictionary obtained from the Wall Street Journal. English tag dictionary is much more ambiguous because it is obtained from a much larger dataset. The random baseline gives an accuracy of around 55%.

The second set of experiments uses more massive parallel data. We used more data to see how our model scales in accuracy and speed. More specifically, we use the news-commentary corpus EUROPARL released by the ACL 2010 Workshop on Statistical Machine Translation [Koehn, 2005]. We used manually annotated test

Language	Random	HMM	MRF
Bulgarian	82.7	88.9	93.5
English	56.2	90.7	87.0
Serbian	83.4	85.1	89.3
Slovene	84.7	87.4	94.5

Table 3: Unsupervised monolingual results with complete tag dictionary on *1984* data.

data from the French treebank [Abeillé et al., 2003], the Penn treebank [Marcus et al., 1994] and data provided as part of the 2006 CoNLL shared task [Buchholz and Marsi, 2006].

5.2 Supervised Verification

As a very primitive comparison, we trained a supervised MRF model to compare to the results of supervised HMMs. In supervised mode, tags, words and word alignments are all given. The training objective maximizes the probability of the given tagging conditioned on the words and the word alignments. This supervised MRF is also called conditional random fields(CRF). The training procedure is also sampling based. The only difference is that there is no need to sample the words because the tags are the only random variables. The CRF and HMM give very close performance with difference in accuracy less than 0.1%. This shows that the CRF is capable of representing an equivalent model represented by the HMM. It also shows that gradient descent with sampling approximation is capable of finding a good model with the weights initialized to all 0s.

5.3 Monolingual POS induction

We trained our model under monolingual setting as a sanity check for our approximate training algorithm. Our model under monolingual mode is exactly the same as the models introduced in Section 2.2. We ran our model on the *1984* data with complete tag dictionary. A comparison between our result and monolingual directed model is shown in Table 3. "Random" is obtained by choosing a random tag for each word according to the tag dictionary. "HMM" is a Bayesian HMM implemented by Snyder et al. [2008]. We also implemented a basic (non-Bayesian) HMM. We trained the HMM with EM and obtained similar results as the Bayesian HMM.

pair	HMM	MRF1	MRF 2
Bulgarian	94.5	93.3	90.5
English	92.0	91.6	91.3
Serbian	91.8	88.1	91.8
Slovene	95.1	87.7	95.0
English	92.0	91.9	92.7
Slovene	88.5	87.8	95.0
Bulgarian	92.0	93.4	90.7
Serbian	86.6	88.7	85.0
English	91.0	89.0	91.6
Serbian	90.1	92.1	89.2
Bulgarian	90.9	90.4	90.2
Slovene	88.2	88.1	88.0

Table 4: Unsupervised bilingual results with complete tag dictionary on 1984 data.

5.4 Bilingual POS induction (1984)

A comparison of unsupervised results between Bayesian HMM and MRF is shown in Table 4. Bayesian HMM is the model built by Snyder et al. [2008]. MRF1 and MRF2 are two runs of our model initialized randomly. Even though the level of ambiguity is low, we can still see oscillations in the range of about 5% in both my model and the Bayesian HMM. The reason as we concluded is that there are a few very common words in the data such as "the," "is" and equivalent words in the other languages. These words are almost always aligned to each other and therefore word alignments are not indicative of the tagging. Labeling these words completely right or completely wrong are both local optima to the model.

By adjusting the weights of a few features, the MRF model can easily switch between these local optima. Flipping the values of those weights during initialization would lead to completely different solutions. The model will be stuck at whatever local optimum it started at. The difference in initialization would eventually lead to significant difference in accuracy. Such effects of initialization for unsupervised models are well known phenomena. For an example, refer to [Johnson, 2007].

The training procedure of the model is tricky to tune. The model is originally trained with stochastic gradient descent with on-line update and a Metropolis Hastings step for sampling the words. It turns out that on-line updates almost always guide the model to a local optimum with low accuracy. The effect of on-line update is very hard to study and is not well-understood. Then we switched to gradient descent with batch updates and the behavior is more regular. We control the step

language pair	Bilingual HMM	Bilingual MRF
English	71.3	72.3
Bulgarian	62.6	62.8
Serbian	54.1	56.2
Slovene	59.7	62.0
English	66.5	73.0
Slovene	53.8	55.5±2.6
Bulgarian	54.2	56.0±1.6
Serbian	56.9	57.0
English	68.2	71.77
Serbian	54.7	57.20
Bulgarian	55.9	59.1±1.1
Slovene	58.5	62.9

Table 5: Unsupervised bilingual results with tag dictionary only for the top 100 frequent words. The standard deviation of a few languages are shown because the differences between each iteration are noticable. The variances exist probably because the step sizes in our gradient descent algorithm are too large.

size by limiting the maximum absolute value of partial derivatives. In this task, regularization seems to only hurt the performance. The magnitude of the weights are already limited by the sampling step.

We also compared the results when only a small portion of the tag dictionary is available. This set of result is more interesting because manually creating a small tag dictionary is more realistic than having a complete tag dictionary. Another reason for using a small tag dictionary is that there is much more for the models to learn compared to the case with complete tag dictionaries. The tag dictionaries only contain the top 100 most frequent words for each language. In English, frequent words include "a," "the," etc. The results are shown in Table 5.

The results are not satisfactory even though they are still comparable to the HMM baseline. The model was much worse when trained with the likelihood objective. To see whether the bad behavior is caused by the training algorithm or the objective function, we tried using exact inference instead of sampling to optimize the same objective. We found that the likelihood objective has lots of bad local optima. The model easily gets stuck in those local optima regardless of the training algorithm. The bad solutions make the model use less tags when more tags are available. In English for example, most of the words are tagged as verbs and nouns. This behavior is the opposite of that of a directed model. A directed model

language pair	Basic Feature	Prefix suffix feature
English	72.1	72.3
Bulgarian	56.2	62.8
Serbian	47.2	56.2
Slovene	52.7	62.0

Table 6: Effect of prefix and suffix features.

tends to use more tags whenever it can.

To make the objective function easier to optimize, we switched to contrastive estimation. The intuition is that word ordering is more important than picking words from the vocabulary for learning syntax of a language. The contrastive objective works well compared to the full objective. The weights learned by the model show that the model is focusing much more on transition features and alignment features rather than emission features. The transition features and alignment features are very powerful for modeling word ordering.

One potential advantage of an undirected model is that it allows arbitrary features. In the case with complete tag dictionaries, we experimented with prefix and suffix but only got worse performance. With more features, the model is more likely to over-fit. Since every word already has a small list of possible tags, prefix and suffix features is not going to help at all. When we switched to using a small portion of the tag dictionary, the performance is very different. A comparison is show in Table 6.

5.5 Crossing Links

Another potential advantage of MRFs is that they allow crossing links. However, in this particular task, crossing links don't make a significant difference. The reason is that these languages are all very similar and there are very few crossing links. They are too few to make a difference. I'm hoping to see a more significant effect with language pairs that have more crossing links. French and English is a promising language pair to look at. There are 87k out of 673k crossing links in our dataset. A comparison of the effect of crossing links is shown in Table 5.5.

5.6 Large-scale corpora

To see how our model scales to larger data sets, we ran our model on the EUROPARL data [Koehn, 2005]. There are about $50,000$ sentences in each language pair. We trained our model with the first $10,000$ sentences for speed consider-

Language	With Crossing links	Without Crossing Link
French	73.8	70.3
English	56.0	59.2

Table 7: Effect of removing crossing links for French and English.

language pair	Random	Basic Feature	Prefix suffix feature
French	63.6	89.8	93.2
English	72.0	90.4	88.7
German	80.4	93.2	93.4
English	72.0	90.8	90.2
Czech	83.3	91.7	93.8
English	72.0	90.1	90.1

Table 8: Tagging accuracies on EUROPARL dataset with complete tag dictionaries.

ations. We experimented both with a complete tag dictionary and with a partial dictionary. The tagging accuracy on 1000 test sentences are shown in Table 8 and 9.

language pair	Random	Basic Feature	Prefix suffix feature
French	44.6	71.8	73.8
English	42.8	57.8	56.0
German	47.3	59.2	59.3
English	42.8	59.8	63.4
Czech	50.7	64.0	63.6
English	43.0	60.0	63.8

Table 9: Tagging accuracies on EUROPARL dataset with reduced tag dictionaries.

6 Error and/or Model analysis

6.1 Why Contrastive

When we first developed the model, we estimated the parameters by maximizing the joint likelihood in Equation 5. In practice, we found that it is very hard to optimize against this objective both in small synthetic cases and in the real data.

In the real data, the model with likelihood objective performed at least 10% worse than with contrastive objective. The tagging it finds is very bad because it uses very few tags. Many less common tags disappeared. The model uses frequent tags for almost all words. It is not learning a useful tagging.

6.2 Synthetic Example

In the synthetic example, we used exact inference to find out that a tagging that uses more tags does correspond to a high likelihood. However, there are many other local optima that correspond to bad taggings. In a bad tagging, some tags just disappear and never get used. This would almost never happen in an HMM because in an HMM, each tag has to distribute its probability mass to some words. The only way to make a tag disappear in an HMM is to make all transition probabilities to this tag very small. But in an MRF, since the parameters are globally optimized, a tag can have tiny probability for all the words in the vocabulary and then not get used.

Contrastive objective gets around this problem by focusing more on transition probabilities than emission probabilities. We compared contrastive objective and length neighborhood objective on synthetic example with exact inference to get rid of any effect due to sampling. Our first synthetic example consists of five sentences where each word is a number. $\{$(0 1 2 3) , (1 2 3 0) , (2 3 0 1) , (3 0 1 2) , (0 1 2 3)$\}$. The MRF model is allowed to use 4 tags. To maximize the data likelihood, the model only has to allocate one tag for each word. There are $4! = 24$ equally good taggings that achieve global optima. As a side note, HMM always uses as many tags as possible with random starts and it is the global optimum for the objective function of HMM. MRF doesn't have such tendency on this synthetic example. Since $\mathbf{w} = \mathbf{0}$ is a local optima, we start our MRF models by randomizing the weights in the range of $[-0.5, 0.5]$. A comparison of histograms of local optima for these two objectives is shown in Figure 5.

When the length of sentences, number of word types and tags are increased to 6 and 7, MRF finds the tagging that uses all tags much less frequently. The global maximum of the contrastive objective does not correspond to the tagging that uses all available tags. Histograms of local optima found by the two models are also

shown in Figure 5. The contrast is more obvious in this case. Contrastive objective is much easier to optimize although its global optimum is to not use all available tags.

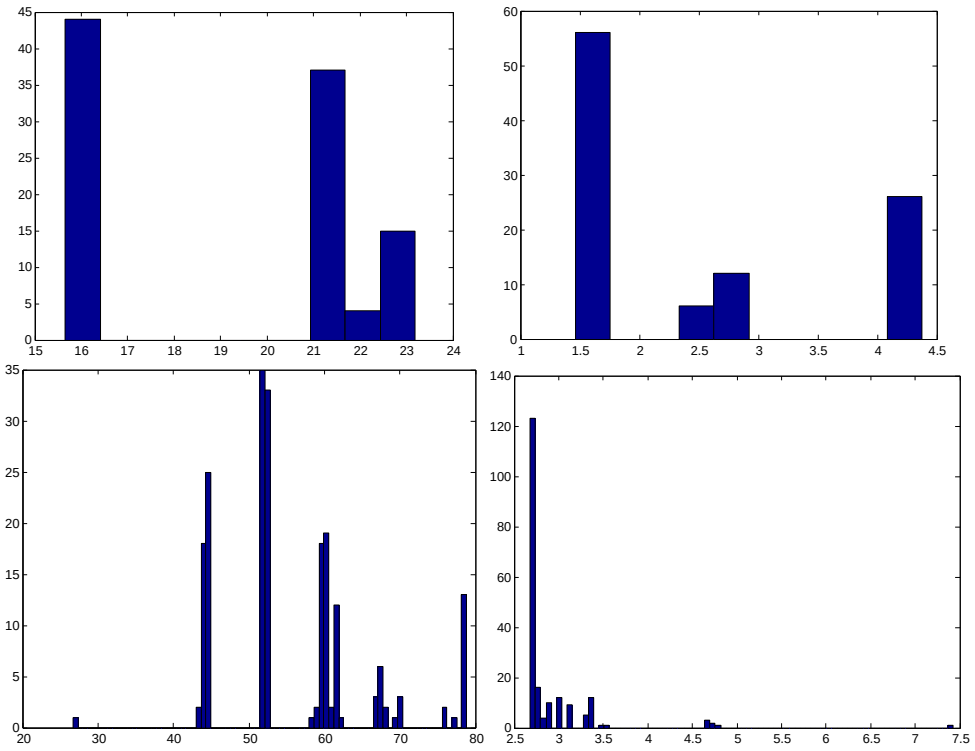

Figure 5: Histograms of local optima found by optimizing the length neighborhood objective (left) and by the contrastive objective (right) on a synthetic dataset with 5 sentences of length 4 (upper row) and 7 (lower row). The axes are frequency versus negation of log-likelihood. A lower value on the horizontal axis corresponds to a better solution.

7 Conclusion

Through a series of experiments, we explored the capabilities of unsupervised MRFs that are not a simple linear chains. The model is more challenging because exact inference with dynamic programming is not applicable. We are forced to experiment with approximate inference technique. In this task, we used Gibbs sampling for simplicity. We obtained comparable results to directed version of the models on the bilingual POS tag induction task [Snyder et al., 2008]. From the results, we believe that there is still room for improvement by picking features more carefully.

MRFs are very flexible models that are tricky to train. We found that contrastive estimation is a useful tool for guiding our model. We experimented with the DynaSearch neighborhood and gained improvement over our original model trained with maximum likelihood objective. By looking at the weights learned by our model, we believe that contrastive estimation drives our model to focus on more general patterns such as transition features and word alignment features. Our experiment with synthetic example shows that it is easier to find the global optima of the contrastive objective with DynaSearch neighborhood on our examples. We still don't fully understand what makes an unsupervised undirected model harder to optimize than a directed model. Looking at histograms is one way of studying this problem. In the future, we may look at this problem from different perspectives and come up with a more concrete answer.

8 Acknowledgments

I would like to thank Chris Dyer and Shay Cohen for their technical expertise and their review of my thesis drafts. They provided many helpful suggestions and ideas during the course of my research. Even though not all ideas worked, all the ideas are very insightful and some of the ideas lead to major developments of the model.

References

Anne Abeillé, Lionel Clément, and François Toussenel. Building a treebank for French. In Anne Abeillé, editor, *Treebanks*. Kluwer, Dordrecht, 2003.

T. Berg-Kirkpatrick, A. Bouchard-Cote, J. DeNero, and D. Klein. Unsupervised learning with features. In *Proceedings of NAACL*, 2010.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

S. Buchholz and E. Marsi. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of CoNLL*, 2006.

G. Casella and E. I. George. Explaining the gibbs sampler. *The American Statistician*, 46 (3):167–174, 1992.

S. B. Cohen and N. A. Smith. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of HLT-NAACL*, 2009.

Richard K. Congram, Chris N. Potts, and Steef L. van de Velde. An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS JOURNAL ON COMPUTING*, 14(1):52–67, 2002. doi: 10.1287/ijoc.14.1.52.7712. URL `http://joc.journal.informs.org/cgi/content/abstract/14/1/52`.

Dipanjan Das and Slav Petrov. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Procedings of ACL 2011*, Portland, OR, June 2011. Association for Computational Linguistics.

Toma Erjavec. Multext-east version 3: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC'04 (ELRA)*, Paris, 2004. International Conference on Language Resources and Evaluation.

Sharon Goldwater and Tom Griffiths. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, Prague, Czech Republic, 2007.

Aria Haghighi and Dan Klein. Prototype-driven learning for sequence models. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 320–327, New York City, USA, June 2006. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/N/N06/N06-1041`.

Mark Johnson. 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, Prague, Czech Republic, 2007.

Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. In *MT Summit 2005*. MT Summit, 2005.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, 2001.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. The Penn treebank: annotating predicate argument structure. In *Proceedings of the workshop on Human Language Technology*, HLT '94, pages 114–119, 1994.

Franz Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Chris N. Potts and Steef L. van de Velde. Dynasearchiterative local improvement by dynamic programming. part i. the traveling salesman problem. *Technical report*, 1995.

N. A. Smith and J. Eisner. Guiding unsupervised grammar induction using contrastive estimation. In *Proceedings of IJCAI Workshop on Grammatical Inference Applications*, 2005a.

Noah A. Smith and Jason Eisner. Contrastive estimation: training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 354–362, 2005b.

Benjamin Snyder, Tahira Naseem, Jacob Eisenstein, and Regina Barzilay. Unsupervised multilingual learning for POS tagging. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1041–1050, Honolulu, Hawaii, 2008.

Benjamin Snyder, Tahira Naseem, Jacob Eisenstein, and Regina Barzilay. Adding more languages improves unsupervised multilingual part-of-speech tagging: a Bayesian nonparametric approach. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 83–91, 2009.

Kristina Toutanova and Mark Johnson. A Bayesian LDA-based model for semi-supervised part-of-speech tagging. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS)*, 2007.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *IN PROCEEDINGS OF HLT-NAACL*, pages 252–259, 2003.