The infinite HMM for unsupervised PoS tagging

Jurgen Van Gael
Department of Engineering
University of Cambridge
jv249@cam.ac.uk

Andreas Vlachos
Computer Laboratory
University of Cambridge
av308@cl.cam.ac.uk

Zoubin Ghahramani
Department of Engineering
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

We extend previous work on fully unsupervised part-of-speech tagging. Using a non-parametric version of the HMM, called the infinite HMM (iHMM), we address the problem of choosing the number of hidden states in unsupervised Markov models for PoS tagging. We implemented a parallelized version of the inference algorithms for the iHMM so we can experiment using the Wall Street Journal dataset and applied two non-parametric priors, Dirichlet Process and Pitman-Yor Process. We evaluate the results with a variety of clustering evaluation metrics and achieve equivalent or better performances than previously reported. Building on this promising result we evaluate the output of the unsupervised PoS tagger as a direct replacement for the output of a fully supervised PoS tagger for the task of shallow parsing and compare the two evaluations.

1 Introduction

Many Natural Language Processing (NLP) tasks are commonly tackled using supervised learning approaches. These learning methods rely on the availability of labeled datasets which are usually produced by expensive manual annotation. For some tasks, we have the choice to use unsupervised learning approaches. While they do not necessarily achieve the same level of performance, they are appealing as unlabeled data is usually abundant. In particular, for the purpose of exploring new domains and languages, obtainining labeled material can be prohibitively expensive and unsupervised learning methods are a very attractive choice. Recent work (Johnson, 2007; Goldwater and Griffiths, 2007; Gao and Johnson, 2008) explored the task of part-of-speech tagging (PoS) using unsupervised Hidden Markov Models (HMMs) with encouraging results. PoS tagging is a standard component in many linguistic processing pipelines, so any improvement on its performance is likely to impact a wide range of tasks.

It is important to point out that a completely unsupervised learning method will discover the statistics of a dataset *according to a particular model choice* but these statistics might not correspond exactly to our intuition about PoS tags. Johnson (2007) and Gao & Johnson (2008) assume that words are generated by a hidden Markov model and find that the resulting states strongly correlate with POS tags. Nonetheless, *identifying* the HMM states with appropriate POS tags is hard. Because many evaluation methods often require POS tags (rather than HMM states) this identification problem makes unsupervised systems difficult to evaluate.

One potential solution is to add a small amount of supervision as in Goldwater & Griffiths (2007) who assume a dictionary of frequent words associated with possible PoS tags extracted from a labeled corpus. Although this technique improves performance, in this paper we explore the completely unsupervised approach. The reason for this is that better unsupervised approaches provide us with better starting points from which to explore how and where to incorporate supervision.

In previous work on unsupervised PoS tagging a main question was how to set the number of hidden states appropriately. Johnson (2007) reports results for different numbers of hidden states but it is unclear how to make this choice a priori, while Goldwater & Griffiths (2007) leave this question as future work.

It is not uncommon in statistical machine learning to distinguish between parameters of a model and the capacity of a model. E.g. in a clustering context, the choice for the number of clusters (capacity) and the parameters of each cluster are often

treated differently: the latter are estimated using algorithms like EM, MCMC or Variational Bayes while the former is chosen using common sense, heuristics or in a Bayesian framework maybe using evidence maximization.

Non-parametric Bayesian methods are a class of probability distributions which explicitly treat the capacity of a model as "just another parameter". Potential advantages are

- the model capacity can automatically adjust to the amount of data: e.g. when clustering a very small dataset, it is unlikely that many fine grained clusters can be distinguished,
- inference can be more efficient: e.g. instead of running full inference for different model capacities and then choosing the best capacity (according to some choice of "best"), inference in non-parametric Bayesian methods integrates the capacity search in one algorithm. This is particularly advantageous when parameters other than capacity need to be explored, since it reduces signifcantly the number of experiments needed.

None of these potential advantages are guaranteed and in this paper we investigate these two aspects for the task of unsupervised PoS tagging.

To summarize, we extend previous work on unsupervised PoS tagging with the following contributions. First, we introduce the use of a non-parametric version of the HMM, namely the infinite HMM (iHMM) (Beal et al., 2002) for unsupervised PoS tagging. This answers an open problem from Goldwater & Griffiths (2007). We carefully implemented a parallelized version of the inference algorithms for the iHMM so we could use it on the Wall Street Journal Penn Treebank dataset. We evaluate the results with a variety of clustering evaluation methods and achieve equivalent or better performances than previously reported. Building on this promising result we use the output of the unsupervised PoS tagger as a direct replacement for the output of a fully supervised PoS tagger for the task of shallow parsing. This evaluation enables us to assess the applicability of an unsupervised PoS tagging method and provides us with means of comparing its performance against a supervised PoS tagger.

The rest of the paper is structured as follows: in section 2 we introduce the iHMM as a non-parametric version of the Bayesian HMM used in previous work on unsupervised PoS tagging. Then, in section 3 we describe some details of our implementation of the iHMM. In section 4 we present a variety of evaluation metrics to compare our results with previous work. Finally, in section 5 we report our experimental results. We conclude this paper with a discussion of ongoing work and experiments.

2 The Infinite HMM

In this section, we describe a non-parametric hidden Markov model known as the infinite HMM (iHMM) (Beal et al., 2002; Teh et al., 2006). As we show below, this model is flexible in the number of hidden states which it can accomodate. In other words, *the capacity is an uncertain quantity* with an a priori infinite range that is a posteriori inferred by the data. It is instructive to first review the finite HMM and its Bayesian treatment: for one, it is the model that has been used in previous work on unsupervised PoS tagging, secondly it allows us to better understand the iHMM.

A finite first-order HMM consists of a hidden state sequence $\boldsymbol{s} = (s_1, s_2, \ldots, s_T)$ and a corresponding observation sequence $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$. Each state variable s_t can take on a finite number of states, say $1 \ldots K$. Transitions between states are governed by Markov dynamics parameterized by the transition matrix $\boldsymbol{\pi}$, where $\pi_{ij} = p(s_t = j | s_{t-1} = i)$, while the initial state probabilities are $\pi_{0i} = p(s_1 = i)$. For each state $s_t \in \{1 \ldots K\}$ there is a parameter ϕ_{s_t} which parameterizes the observation likelihood for that state: $y_t | s_t \sim F(\phi_{s_t})$. Given the parameters $\{\pi_0, \boldsymbol{\pi}, \boldsymbol{\phi}, K\}$ of the HMM, the joint distribution over hidden states $\boldsymbol{s}$ and observations $\boldsymbol{y}$ can be written (with $s_0 = 0$):

$$p(\boldsymbol{s}, \boldsymbol{y} | \pi_0, \boldsymbol{\pi}, \boldsymbol{\phi}, K) = \prod_{t=1}^{T} p(s_t | s_{t-1}) p(y_t | s_t)$$

As Johnson (2007) clearly explained, training the HMM with EM leads to poor results in PoS tagging. However, we can easily treat the HMM in a fully Bayesian way (MacKay, 1997) by introducing priors on the parameters of the HMM. With no further prior knowledge, a typical prior for the transition (and initial) probabilities are symmetric Dirichlet distributions. This corresponds to our belief that, a priori, each state is equally likely to transition to every other state. Also, it is commonly known that the parameter of a Dirichlet

distribution controls how sparse its samples are. In other words, by making the hyperprior on the Dirichlet distribution for the rows of the transition matrix small, we can encode our belief that any state (corresponding to a PoS tag in this application context) will only be followed by a small number of other states. As we explain below, we will be able to include this desirable property in the non-parametric model as well. Secondly, we need to introduce a prior on the observation parameters ϕ_k. Without any further prior knowledge, a convenient choice here is another symmetric Dirichlet distribution with sparsity inducing hyperprior. This encodes our belief that only a subset of the words correspond to a particular state.

A first naïve way to obtain a non-parametric HMM with an infinite number of states might be to use symmetric Dirichlet priors over the transition probabilities with parameter α/K and take $K \to \infty$. This approach unfortunately does not work: $\alpha/K \to 0$ when $K \to \infty$ and hence the rows of the matrix will become "infinitely sparse". Since the sum of the entries must sum to one, the rows of the transition matrix will be zero everywhere and all its mass in a random location. Unfortunately, this random location is out of an infinite number of possible locations and hence with probability 1 will be different for all the rows. As a consequence, at each timestep the HMM moves to a new state and will never revisit old states. As we shall see shortly, we can fix this by using a hierarchical Bayesian formalism where the Dirichlet priors on the rows have a shared parameter.

Before moving on to the iHMM, let us look at the finite HMM from a different perspective. The finite HMM of length T with K hidden states can be seen as a sequence of T finite mixture models. The following equation illustrates this idea: conditioned on the previous state s_{t-1}, the marginal probability of observation y_t can be written as:

$$\begin{aligned} p(y_t|s_{t-1}=k) &= \sum_{s_t=1}^{K} p(s_t|s_{t-1}=k)p(y_t|s_t), \\ &= \sum_{s_t=1}^{K} \pi_{k,s_t} p(y_t|\phi_{s_t}). \end{aligned} \quad (1)$$

The variable $s_{t-1} = k$ specifies the mixing weights $\pi_{k,\cdot}$ for the mixture distribution, while s_t indexes the mixture component generating the observation y_t. In other words, equation (1) says that each row of the transition matrix $\boldsymbol{\pi}$ specifies a different mixture distribution over the same set of K mixture components $\boldsymbol{\phi}$.

Our second attempt to define a non-parametric version of the hidden Markov model is to replace the finite mixture by an infinite mixture. The theory of Dirichlet process mixtures (Antoniak, 1974) tells us exactly how to do this. A draw $G \sim DP(\alpha, H)$ from a Dirichlet process (DP) with base measure H and concentration parameter α is a discrete distribution which can be written as an infinite mixture of atoms

$$G(\cdot) = \sum_{l=1}^{\infty} \pi_l \delta_{\phi_l}(\cdot)$$

where the ϕ_l are i.i.d. draws from the base measure H, $\delta_{\phi_l}(\cdot)$ represents a point distribution at ϕ_l and π follows a infinite dimensional version of the Dirichlet distribution. We refer to Teh et al. (2006) for more details.

Switching back to the iHMM our next step is to introduce a DP G_j for each state $j \in \{1 \cdots \infty\}$; we write $G_j(\cdot) = \sum_{i=1}^{\infty} \pi_{ji} \delta_{\phi_{ji}}(\cdot)$. There is now a parameter for each state j and each index $i \in \{1, 2, \cdots, \infty\}$. Next, we draw the datapoint at timestep t given that the previous datapoint was in state s_{t-1} by drawing from DP $G_{s_{t-1}}$. We first select a mixture component s_t from the vector $\pi_{s_{t-1},\cdot}$ and then sample a datapoint $y_t \sim F(\phi_{s_{t-1},s_t})$ so we get the following distribution for y_t

$$p(y_t|\alpha, s_{t-1}) = \sum_{s_t=1}^{\infty} \pi_{s_{t-1},s_t} p(y_t|\phi_{s_{t-1},s_t}).$$

This is almost the non-parametric equivalent of equation (1) but there is a subtle difference: each DP G_j selects their own set of parameters $\phi_{j\cdot}$. This is unfortunate as it means that the output distribution would not be the same for each state, it would depend on which state we were moving to! Luckily, we can easily fix this: by introducing an intermediate distribution $G_0 \sim DP(\gamma, H)$ and let $G_j \sim DP(\alpha, G_0)$ we enforce that the i.i.d. draws $\phi_{j\cdot}$ are draws from a discrete distribution (since G_0 is a draw from a Dirichlet process) and hence all G_j will share the same infinite set of atoms as chosen by G_0. Figure 1 illustrates the graphical model for the iHMM.

Hyperparameter Choice The description above shows that there are 4 parameters which we must specify: the base measure H, the

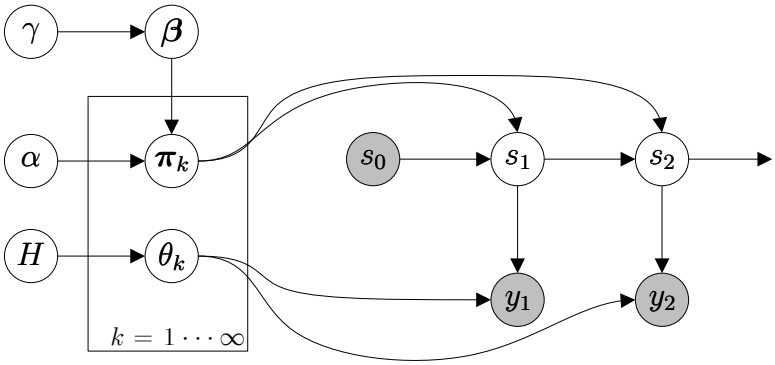

Figure 1: The graphical model for the iHMM. The variable β represents the mixture for the DP G_0.

output distribution $p(y_t|\phi_{s_t})$, the concentration parameter γ for DP G_0 and the concentration parameter α for the DP's G_j. Just as in the finite case, the base measure H is the prior distribution on the parameter ϕ of $p(y_t|\phi_{s_t})$. We chose to use a symmetric Dirichlet distribution with parameter δ over the word types in our corpus. Since we do not know the sparsity level δ of the output distributions we decided to learn this parameter from the data. We initially set a vague Gamma prior over δ but soon realized that as we expect hidden states in the iHMM to correspond to PoS tags, it is unrealistic to expect each state to have the same sparsity level. Hence we chose a Dirichlet process as the prior for δ; this way we end up with a small discrete set of sparsity levels: e.g. we can learn that states corresponding to verbs and nouns share one sparsity level while states correpsonding to determiners have their own (much sparser) sparsity level. For the output distribution $p(y_t|\phi_{s_t})$ we chose a simple multinomial distribution.

The DP hyperparameter γ mostly controls the number of states in the iHMM while - as we discussed above - α controls the sparsity of the transition matrix. In the experiments below we report both fixing the two parameters and learning them by sampling (using vague Gamma hyperpriors).

3 Inference

The Wall Street Journal part of the Penn Treebank that was used for our experiments contains about one million words. In the non-parametric Bayesian literature not many algorithms have been described that scale into this regime. In this section we describe our implementation of the iHMM which as far as we know is the first that tackles datasets of this scale.

There is a wealth of evidence (Scott, 2002; Gao and Johnson, 2008) in the machine learning literature that Gibbs sampling for Markov models leads to slow mixing times. Hence we decided our starting point for inference needs to be based on dynamic programming. Because we didn't have a good idea for the number of states that we were going to end up with, we prefered the beam sampler of Van Gael et al. (2008) over a finite truncation of the iHMM. Moreover, the beam sampler also introduces a certain amount of sparsity in the dynamic program which can speed up computations (potentially at the cost of slower mixing).

The beam sampler is a blocked Gibbs sampler where we alternate between sampling the parameters (transition matrix, output parameters), the state sequence and the hyperparameters. Sampling the transition matrix and output distribution parameters requires computing their sufficient statistics and sampling from a Dirichlet distribution; we refer to the beam sampling paper for details. For the hyperparameters we use standard Gibbs sampling. We briefly sketch the resampling step for the state sequence for a single sequence of data (sentence of words). Running standard dynamic programming is prohibitive because the state space of the iHMM is infinitely large. The central idea of the beam sampler is to adaptively truncate the state space of the iHMM and run dynamic programming. In order to truncate the state space, we sample an auxilary variable u_t for each word in the sequence from the distribution $u_t \sim \mathsf{Uniform}(0, \pi_{s_{t-1}s_t})$ where π represents the transition matrix.

Intuitively, when we sample $u_{1:T}|s_{1:T}$ according to the distribution above, the only valid sam-

ples are those for which the u_t are smaller than the transition probabilities of the state sequence $s_{1:T}$. This means that when we sample $s_{1:T}|u_{1:T}$ at a later point, it must be the case that the u_t's are still smaller than the new transition probabilities. This significantly reduces the set of valid state sequences that we need to consider. More specifically, Van Gael et al. (2008) show that we can compute $p(s_t|y_{1:t}, u_{1:t})$ using the following dynamic programming recursion $p(s_t|y_{1:t}, u_{1:t}) =$

$$p(y_t|s_t) \sum_{s_{t-1}:u_t<\pi_{s_{t-1},s_t}} p(s_{t-1}|y_{1:t-1}, u_{1:t-1}).$$

The summation $\sum_{s_{t-1}:u_t<\pi_{s_{t-1},s_t}}$ ensures that this computation remains finite. When we compute $p(s_t|y_{1:t}, u_{1:t})$ for $t \in \{1 \cdots T\}$, we can easily sample s_T and using Bayes rule backtrack sample every other s_t. It can be shown that this procedure produces samples from the exact posterior.

Notice that the dynamic program only needs to perform computation when $u_t < \pi_{s_{t-1},s_t}$. A careful implementation of the beam sampler consists of preprocessing the transition matrix π and sorting its elements in descending order. We can then iterate over the elements of the transition matrix starting from the largest element and stop once we reach the first element of the transition matrix smaller than u_t. In our experiments we found that this optimization reduces the amount of computation per sentence by an order of magnitutde.

A second optimization which we introduced is to use the map-reduce paradigm (Dean and Ghemawat, 2004) to parallelize our computations. More specifically, after we preprocess the transition matrix, the dynamic program computations are independent for each sentence in the dataset. This means we can perform each dynamic program in parallel; in other words our "map" consists of running the dynamic program on one sentence in the dataset. Next, we need to resample the transition matrix and output distribution parameters. In order to do so we need to compute their sufficient statistics: the number of transitions from state to state and the number of emissions of each word out of each state. Our "reduce" function consists of computing the sufficient statistics for each sentence and then aggregating the statistics for the whole dataset. Our implementation runs on a quad-core shared memory architecture and we find an almost linear speedup going from one to four cores.

4 Evaluation

Evaluating unsupervised PoS tagging is rather difficult mainly due to the fact that the output of such systems are not actual PoS tags but state identifiers. Therefore it is impossible to evaluate performance against a manually annotated gold standard. Recent work (Goldwater and Griffiths, 2007; Johnson, 2007; Gao and Johnson, 2008) on this task explored a variety of methodologies to address this issue.

The most common approach followed in previous work is to evaluate unsupervised PoS tagging as clustering against a gold standard using the Variation of Information (VI) (Meilă, 2007). VI assesses homogeneity and completeness using the quantities $H(C|K)$ (the conditional entropy of the class distribution in the gold standard given the clustering) and $H(K|C)$ (the conditional entropy of clustering given the class distribution in the gold standard). However, as Gao & Johnson (2008) point out, VI is biased towards clusterings with a small number of clusters. A different evaluation measure that uses the same quantities but weighs them differently is the V-measure (Rosenberg and Hirschberg, 2007), which is defined in Equation 2 by setting the parameter β to 1.

$$\begin{aligned} h &= 1 - \frac{H(C|K)}{H(C)} \\ c &= 1 - \frac{H(K|C)}{H(K)} \\ V_\beta &= \frac{(1+\beta)hc}{(\beta h) + c} \end{aligned} \tag{2}$$

Vlachos et al. (2009) noted that V-measure favors clusterings with a large number of clusters. Both of these biases become crucial in our experiments, since the number of clusters (states of the iHMM) is not fixed in advance. Vlachos et al. proposed a variation of the V-measure, V-beta, that adjusts the balance between homogeneity and completeness using the parameter β in Eq. 2.

It is worth mentioning that, unlike V-measure and V-beta, VI scores are not normalized and therefore they are difficult to interpret. Meilă (2007) presented two normalizations, acknowledging the potential disadvantages they have. The first one normalizes VI by $2\log(max(|K|, |C|))$, which is inappropriate when the number of clusters discovered $|K|$ changes between experiments. The second

normalization involves the quantity $\log N$ which is appropriate when comparing different algorithms on the same dataset (N is the number of instances). However, this quantity depends exclusively on the size of the dataset and hence if the dataset is very large it will result in a normalized VI score misleadingly close to 100%. This does not affect rankings, i.e. a better VI score will also be translated into a better normalized VI score. In our experiments, we report results only with the un-normalized VI scores, V-measure and V-beta.

All the evaluation measures mentioned so far evaluate PoS tagging as a clustering task against a manually annotated gold standard. While this is reasonable, it still does not provide means of assessing the performance in a way that would allow comparisons with supervised methods that output actual PoS tags. Even for the normalized measures V-measure and V-beta, it is unclear how their values relate to accuracy levels. Gao & Johnson (2008) partially addressed this issue by mapping states to PoS tags following two different strategies, cross-validation accuracy, and greedy 1-to-1 mapping, which both have shortcomings. We argue that since an unsupervised PoS tagger is trained without taking any gold standard into account, it is not appropriate to evaluate against a particular gold standard, or at least this should not be the sole criterion. The fact that different authors use different versions of the same gold standard to evaluate similar experiments (e.g. Goldwater & Griffiths (2007) versus Johnson (2007)) supports to this claim. Furthermore, PoS tagging is seldomly a goal in itself, but it is a component in a linguistic pipeline.

In order to address these issues, we perform an extrinsic evaluation using a well-explored task that involves PoS tags. While PoS tagging is considered a pre-processing step in many natural language processing pipelines, the choice of task is restricted by the lack of real PoS tags in the output of our system. For our purposes we need a task that relies on discriminating between PoS tags rather than the PoS tag semantics themselves, in other words, a task in which knowing whether a word is tagged as noun instead of a verb is equivalent to knowing it is tagged as state 1 instead of state 2. Taking these considerations into account, in Section 5 we experiment with shallow parsing in the context of the CoNLL-2000 shared task (Tjong Kim Sang and Buchholz, 2000) in which very good performances were achieved using only the words with their PoS tags. Our intuition is that if the iHMM (or any unsupervised PoS tagging method) has a reasonable level of performance, it should improve on the performance of a system that does not use PoS tags. Moreover, if the performance is very good indeed, it should get close to the performance of a system that uses real PoS tags, provided either by human annotation or by a good supervised system. Similar extrinsic evaluation was performed by Biemann et al. (Biemann et al., 2007). It is of interest to compare the results between the clustering evaluation and the extrinsic one.

A different approach in evaluating non-parametric Bayesian models for NLP is state-splitting (Finkel et al., 2007; Liang et al., 2007). In this setting, the model is used in order to refine existing annotation of the dataset. While this approach can provide us with some insights and interpretable results, the use of existing annotation influences the output of the model. In this work, we want to verify whether the output of the iHMM (without any supervision) can be used instead of that of a supervised system.

5 Experiments

In all our experiments, the Wall Street Journal (WSJ) part of the Penn Treebank was used. As explained in Section 4, we evaluate the output of the iHMM in two ways, as clustering with respect to a gold standard and as direct replacement of the PoS tags in the task of shallow parsing. In each experiment, we obtain a sample from the iHMM over all the sections of WSJ. The states for sections 15-18 and 20 of the WSJ (training and testing sets respectvely in the CoNLL shared task) are used for the evaluation based on shallow parsing, while the remaining sections are used for evaluation against the WSJ gold standard PoS tags using clustering evaluation measures.

As described in Section 2 we performed three runs with the iHMM: two runs with fixed hyperparameters α and γ one using the DP prior and one with the PY prior, and one where we learn them from the data using the DP prior. Our inference algorithm uses 1000 burn-in iterations after which we collect a sample every 1000 iterations. Our inference procedure is also tempered during the first 1000 burnin and 2400 iterations. The temper-

ing consists of reducing the influence of the output distribution by powering it with a number that smoothly increases from 0.4 to 1.0 over the 3400 first iterations. The numbers of iterations reported in the remainder of the section refer to the iterations after burn-in. We point out that one weakness of MCMC methods is that they are hard to test for convergence. Our approach was to run the simulations until became prohibitively expensive to obtain the next sample.

First, we present results using clustering evaluation measures which appear in the figures of Table 1. The three runs exhibit different behavior. The number of states reached by the iHMM with fixed parameters using the DP prior stabilizes around 53, while for the experiment with learnt hyperparameters the number of states grows more rapidly, reaching 194 states after 5,000 iterations. With the PY prior, the number of states reached grows less rapidly reaching 90 states. All runs achieve better performances with respect to all the measures used as the number of iterations grows. An exception is that VI scores tend to increase (lower VI scores are better) when the number of states grows larger than the gold standard. It is interesting to notice how the measures exhibit different biases, in particular that VI penalizes the larger numbers of states discovered in the DP run with learnt parameters as well as the run with the PY prior, compared to the more lenient scores provided by V-measure and V-beta. The latter though assigns lower scores to the DP run with learnt parameters because it takes into account that the high homogeneity is achieved using even more states. Finally, the interpretability of these scores presents some interest. For example, in the run with fixed parameters using the DP prior, after burn-in VI was 4.6, which corresponds to 76.65% normalized VI score, while V-measure and V-beta were 12.7% and 9% respectively. In 8,000 iteration after burn-in, VI was 3.94 (80.3% when normalized), while V-measure and V-beta 53.3%, since the number of states was almost the same as the number of unique PoS tags in the gold standard.

The closest experiment to ours is the one by Gao & Johnson (2008) who run their Bayesian HMM over the whole WSJ and evaluated against the full gold standard, the only difference being is that we exclude the CoNLL shared task sections from our evaluation, which leaves us with 19 sections instead of 24. Their best VI score was 4.03886 which they achieved using the collapsed, sentence-blocked Gibbs sampler with the number of states fixed to 50. The VI score achieved by the iHMM with fixed parameters using the PY prior reaches 3.73, while using the DP prior VI reaches 4.32 with learnt parameters and 3.93 with fixed parameters. These results, even if they are not directly comparable, are on par with the state-of-the-art, which encouraged us to proceed with the extrinsic evaluation.

For the experiments with shallow parsing we used the CRF++ toolkit[1] which has an efficient implementation of the model introduced by Sha & Pereira (2003) for this task. First we ran an experiment using the words and the PoS tags provided in the shared task data and the performances obtained were 96.07% accuracy and 93.81% F-measure. The PoS tags were produced using the Brill tagger (Brill, 1994) which employs tranformation-based learning and was trained using the WSJ corpus. Then we ran an experiment removing the PoS tags altogether, and the performances were 93.25% accuracy and 88.58% F-measure respectively. This gave us some indication as to what the contribution of the PoS tags is in the context of a shallow parsing task.

The experiments using the output of the iHMM as PoS tags for shallow parsing are presented in Table 2. The best performance achieved was 94.48% and 90.98% in accuracy and F-measure, which is 1.23% and 2.4% better respectively than just using words, but worse by 1.57% and 2.83% compared to using the supervised PoS tagger output. Given that the latter is trained on WSJ we believe that this is a good result. Interestingly, this was obtained by using the last sample from the iHMM run using the DP prior with learnt parameters which has worse overall clustering evaluation scores, especially in terms of VI. This sample though has the best homogeneity score (69.39%). We believe that homogeneity is more important than the overall clustering score due to the fact that, in the application considered, it is probably worse to assign tokens that belong to different PoS tags to the same state, e.g. verb and adverbs, rather than generate more than one state for the same PoS. This is likely to be the case in tasks where we are interested in distinguishing between PoS tags rather than the actual tag itself. Also, clustering evaluation measures tend to score leniently

[1] http://crfpp.sourceforge.net/

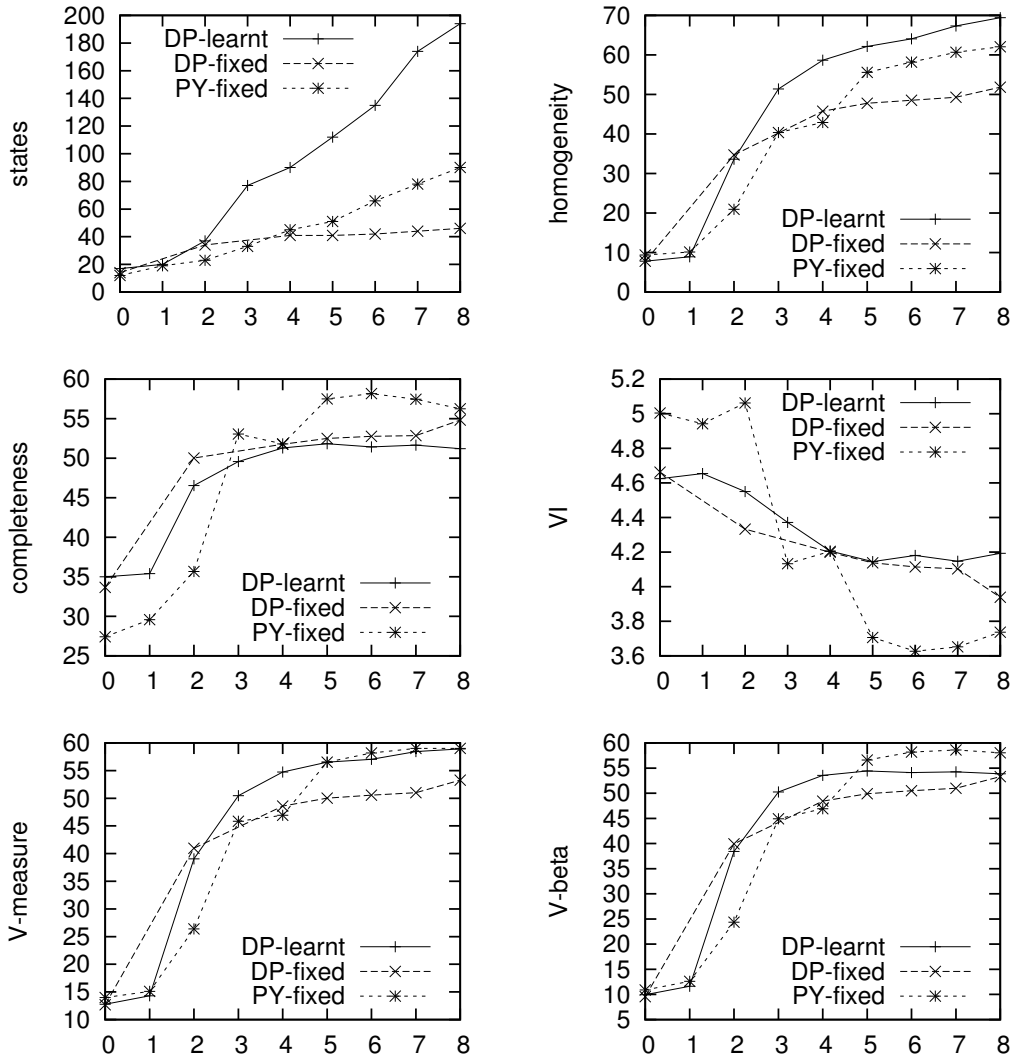

Table 1: Performance of the three iHMM runs according to clustering evaluation measures against number of iteretions (in thousands).

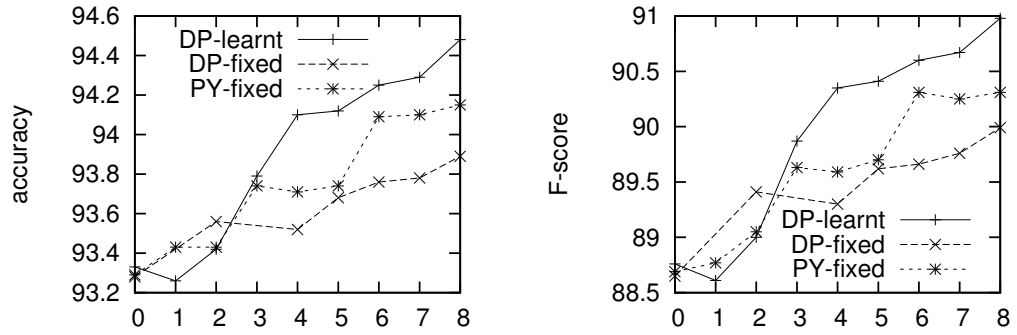

Table 2: Performance of the output of the three iHMM runs when used in shallow parsing against number of iteretions (in thousands).

consistent mixing of members of different classes in the same cluster. However, such mixing results in consistent noise when the clustering output becomes input to a machine learning method, which is harder to deal with.

6 Conclusions - Future Work

In the context of shallow parsing we saw that the performance of the iHMM does not match the performance of a supervised PoS tagger but does lead to a performance increase over a model using only words as features. Given that it was constructed without any need for human annotation, we believe this is a good result. At the same time though, it suggests that it is still some way from being a direct drop-in replacement for a supervised method. We argue that the extrinsic evaluation of unsupervised PoS tagging performed in this paper is quite informative as it allowed us to assess our results in a more realistic context. In this work we used shallow parsing for this, but we are considering other tasks in which we hope that PoS tagging performance will be more crucial.

We are currently experimenting with a semi-supervised PoS tagger where we let the transition matrix of the iHMM depend on annotated PoS tags. This model allows us to: a) use annotations whenever they are available and do unsupervised learning otherwise; b) use the power of non-parametric methods to possibly learn more fine grained statistical structure than tag sets created manually.

On the implementation side, it would be interesting to see how our methods scale in a distributed map-reduce architecture where network communication overhead becomes an issue.

Finally, the ultimate goal of our investigation is to do unsupervised PoS tagging using web-scale datasets. Although the WSJ corpus is reasonably sized, our computational methods do not currently scale to problems with one or two order of magnitude more data. We will need new breakthroughs to unleash the full potential of unsupervised learning for PoS tagging.

References

Charles E. Antoniak. 1974. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174.

M. J. Beal, Z. Ghahramani, and C. E. Rasmussen. 2002. The Infinite Hidden Markov Model. *Advances in Neural Information Processing Systems*, 14:577 – 584.

Chris Biemann, Claudio Giuliano, and Alfio Gliozzo. 2007. Unsupervised part-of-speech tagging supporting supervised methods. In *Proceedings of RANLP*.

Eric Brill. 1994. Some advances in transformation-based part of speech tagging. In *National Conference on Artificial Intelligence*, pages 722–727.

Jeffrey Dean and Sanjay Ghemawat. 2004. Mapreduce: Simplified data processing on large clusters. In *Sixth Symposium on Operating System Design and Implementation*.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. 2007. The infinite tree. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 272–279, Prague, Czech Republic, June. Association for Computational Linguistics.

J. Van Gael, Y. Saatci, Y. W. Teh, and Z. Ghahramani. 2008. Beam Sampling for the Infinite Hidden Markov Model. In *Proceedings of the 25th International Conference on Machine learning*, Helsinki.

Jianfeng Gao and Mark Johnson. 2008. A comparison of Bayesian estimators for unsupervised Hidden Markov Model POS taggers. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 344–352.

Sharon Goldwater and Tom Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 744–751, Prague, Czech Republic, 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*, pages 296–305.

P. Liang, S. Petrov, M. I. Jordan, and D. Klein. 2007. The infinite PCFG using hierarchical Dirichlet processes. In *Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP/CoNLL)*.

D. J. C. MacKay. 1997. *Ensemble learning for hidden Markov models*. Technical report, Cavendish Laboratory, University of Cambridge, 1997.

Marina Meilă. 2007. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 410–420, Prague, Czech Republic, June.

Steven L. Scott. 2002. Bayesian Methods for Hidden Markov Models: Recursive Computing in the 21st Century. *Journal of the American Statistical Association*, 97(457):337–351, March.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Human Language Technology Conference and the 4th Meeting of the North American Association for Computational Linguistics*.

Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and D. M. Blei. 2006. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In Claire Cardie, Walter Daelemans, Claire Nedellec, and Erik Tjong Kim Sang, editors, *Proceedings of the Fourth Conference on Computational Natural Language Learning*, pages 127–132. Lisbon, Portugal, September.

Andreas Vlachos, Anna Korhonen, and Zoubin Ghahramani. 2009. Unsupervised and Constrained Dirichlet Process Mixture Models for Verb Clustering. In *Proceedings of the EACL workshop on GEometrical Models of Natural Language Semantics*.