Unsupervised Bayesian Part of Speech Inference with Particle Gibbs

Gregory Dubbin and Phil Blunsom

Department of Computer Science, University of Oxford, United Kingdom
Gregory.Dubbin@wolfson.ox.ac.uk
Phil.Blunsom@cs.ox.ac.uk

Abstract. As linguistic models incorporate more subtle nuances of language and its structure, standard inference techniques can fall behind. These models are often tightly coupled such that they defy clever dynamic programming tricks. Here we demonstrate that Sequential Monte Carlo approaches, i.e. particle filters, are well suited to approximating such models. We implement two particle filters, which jointly sample either sentences or word types, and incorporate them into a Particle Gibbs sampler for Bayesian inference of syntactic part-of-speech categories. We analyze the behavior of the samplers and compare them to an exact block sentence sampler, a local sampler, and an existing heuristic word type sampler. We also explore the benefits of mixing Particle Gibbs and standard samplers.

1 Introduction

Modern research is steadily revealing more of the subtle structure of natural language to create increasingly intricate models. Recent advances in Bayesian non-parametrics have been employed by Computational Linguistics researchers to create effective unsupervised models of the latent structure text [1–4]. These models predominantly make use of the Dirichlet Process (DP), and its generalization the Pitman-Yor Process (PYP), in part due to the ease of deriving a collapsed Gibbs sampler for its inference. However this ease of inference comes at a cost; collapsed Gibbs samplers mix poorly due to the tight global dependencies present in the conditional distributions sampled from [5].

Previously [5] investigated techniques for alleviating the influence of long range conditional dependencies by simultaneously sampling groups of latent variables, however this approach is specific to models based on the simpler Dirichlet distribution. Here we present a general inference approach based on Sequential Monte Carlo (SMC) suitable for more advance models that employ Pitman-Yor Process priors.

Sequential Monte Carlo (SMC) methods, like particle filters, are particularly well suited to estimating tightly coupled distributions [6]. Particle filters sample sequences of latent variable assignments by concurrently generating several representative sequences consistent with a model's conditional probability distribution. The sequential nature of the sampling simplifies inference by ignoring ambiguous correlations with future latent variables at the cost of sampling the sequence multiple times. The few applications of particle filters in computational linguistics generally focus on the online nature of SMC [7, 8]. However, in this paper we demonstrate that batch applications

benefit from the power of SMC to generate samples from tightly coupled distributions that would otherwise need to be approximated. Furthermore, the time cost of the additional samples generated by SMC can be mitigated by generating them in parallel.

In this paper we focus on the task of unsupervised part-of-speech (PoS) induction, where we seek to label tokens in a corpus with the syntactic role they play. In particular we describe and evaluate novel SMC inference algorithms for the Pitman-Yor Hidden Markov Model (PYP-HMM) first proposed in [4]. This model represents the state-of-the-art for unsupervised PoS induction, but the complex conditional dependencies present in the posterior led the authors of [4] to resort to a heuristic inference startegy and unrealistic restrictions on the model to achieve their highest reported results.

We start in Section 2 by introducing PYP-HMM model and its previously proposed inference algorithms. Section 3 introduces the Sequential Importance Sampling (SIS) algorithm, a basic SMC method that generates samples from the model's posterior. Using this approach we describe two novel inference algorithms for the PYP-HMM: a simple sentence-based block sampler (3.1) and a more complicated type-based sampler (3.2). We evaluate these algorithms in Section 4, analyzing their behavior in comparisons to the previously proposed state-of-the-art approaches. In summary we show that our SMC based algorithms are able to improve inference, finding higher probability modes, and task specific accuracies.

2 The Pitman-Yor Hidden Markov Model

The PYP-HMM model of PoS induction exhibits the tightly coupled correlations that complicate many standard inference methods [4]. The model applies a hierarchical Pitman-Yor process (PYP) prior to a trigram hidden Markov model (HMM) to jointly model the distribution of a sequence of latent word tags, $\mathbf{t}$, and word tokens, $\mathbf{w}$. The joint probability defined by the transition, $P_\theta(t_l|t_{n-1}, t_{n-2})$, and emission, $P_\theta(w_n|t_n)$, distributions of a trigram HMM is

$$P_\theta(\mathbf{t}, \mathbf{w}) = \prod_{n=1}^{N+1} P_\theta(t_l|t_{n-1}, t_{n-2}) P_\theta(w_n|t_n)$$

where $N = |\mathbf{t}| = |\mathbf{w}|$ and the special tag \$ is added to denote the sentence boundaries. The model defines the transition and emission distributions to be multinomial:

$$\begin{aligned} t_n|t_{n-1}, t_{n-2}, T &\sim T_{t_{n-1},t_{n-2}} \\ w_n|t_n, E &\sim E_{t_n} \end{aligned}$$

The PYP-HMM draws the above multinomial distributions from a hierarchical Pitman-Yor Process prior. The hierarchical prior can be intuitively understood to smooth the trigram transition distributions with bigram and unigram distributions in a similar manner to an ngram language model [9]. This backoff structure greatly reduces sparsity in the trigram distributions and is achieved by chaining together the PYPs through their

base distributions:

$$
\begin{aligned}
T_{ij}|a^T, b^T, B_i &\sim \mathrm{PYP}(a^T, b^T, B_i) \\
B_i|a^B, b^B, U &\sim \mathrm{PYP}(a^B, b^b, U) \\
U|a^U, b^U &\sim \mathrm{PYP}(a^U, b^U, \mathrm{Uniform}). \\
E_i|a^E, b^E, C &\sim \mathrm{PYP}(a^E, b^E, C_i),
\end{aligned}
$$

where T_{ij}, B_i, and U are trigram, bigram, and unigram transition distributions respectively and C_i is either a uniform distribution (PYP-HMM) or a bigram character language model distribution (PYP-HMM+LM, intended to model basic morphology).

Draws from the posterior of the hierarchical PYP can be calculated with a variant of the Chinese Restaraunt Process (CRP) called the Chinese Restaurant Franchise (CRF) [9, 1]. In the CRP analogy, each latent variable (tag) in a sequence is represented by a customer entering a restaurant and sitting at one of an infinite number of tables. A customer chooses to sit at a table in a restaurant according to the probability

$$
P(z_n = k|\mathbf{z}_{1:n-1}) = \begin{cases} \frac{c_k^- - a}{n-1+b} & 1 \leq k \leq K^- \\ \frac{K^- a + b}{n-1+b} & k = K^- + 1 \end{cases} \tag{1}
$$

where z_n is the index of the table chosen by the nth customer to the restaurant, $\mathbf{z}_{1:n-1}$ is the seating arrangement of the previous $n-1$ customers to enter, c_k^- is the count of the customers at table k, and K^- is the total number of tables chosen by the previous $n-1$ customers. All customers at a table share the same dish, representing the value assigned to the latent variables. When customers sit at an empty table, a new dish is assigned to that table according to the base distribution of the PYP. To expand the CRP analogy to the CRF for hierarchical PYPs, when a customer sits at a new table, a new customer enters the restaurant representing the PYP of the base distribution.

Blunsom and Cohn [4] explored two Gibbs sampling methods for inference with the PYP-HMM model. The first individually samples tag assignments for each token. The second employs a tactic shown to be effective by earlier works by constraining inference to only one tag per word type (PYP-1HMM). However marginalizing over all possible table assignments for more than a single tag is intractable, and must be approximated. Blunsom and Cohn [4] approximates the PYP-1HMM tag assignment probabilities for a particular sample according to heuristic fractional table counts. Specific details of this model and associated samplers can be found in [4].

In this paper we present a principled SMC based sampler that allows for the simultaneous sampling of all the tags associated with a given word type without resorting to heuristics. This type based sampler achieves state-of-the-art performance without the requirement that all words of a given type share the same tag. This one-tag-per-type assumption is clearly false for syntactic categories (e.g. *sample* as a verb and a noun), thus its elimination is a step on the path to high performance unsupervised PoS induction.

3 Sequential Monte Carlo

Sequential Monte Carlo was introduced in 1993 as a Bayesian estimator for signal processing problems with strong non-linear conditional dependencies [10]. Since then,

SMC methods have been adopted by many fields, including statistics, biology, economics, etc. [11–13]. The SMC approach is the probabilistic analogue of the beam search heuristic, where the beam width can be compared to the number of particles and pruning is analogous to resampling. The basic SMC approach serves as the basis for several variants. Many SMC implementations resample the population of particles from the existing population to minimize the effect of increasing sample variance with increasing sequence length [14]. Particle smoothing variants of SMC reduce the relative variance of marginals early in the sequence, as well improving the diversity of the final sample [15]. Particle Markov chain Monte Carlo (PMCMC) formally augments classic Markov chain Monte Carlo (MCMC) approaches, like Gibbs sampling, with samples generated by particle filters [6].

While MCMC approximates a distribution as the average of a sequence of samples taken from the posterior of the distribution, SMC approximates a distribution as the importance weighted sum of several sequentially generated samples, called particles. This article describes two SMC samplers that jointly sample multiple tag assignments: a sentence based block sampler (`sent`) and a word type based block sampler (`type`). The basics of particle filtering are outlined below, while the implementation specifics of the `sent` and `type` particle filters are described in secions 3.1 and 3.2, respectively.

SMC is essentially the probabilistic analogue of the beam search heuristic. SMC stores P sequences, analogous to beam width, and extends each incrementally according to a proposal distribution q_n, similar to the heuristic cost function in beam search. Many particle filtering implementations also include a resampling step which acts like pruning by reducing the number of unlikely sequences.

We implemented Sequential Importance Sampling (SIS), detailed by Doucet and Johansen [16], to approximate joint samples from the sentence and word type distributions. This approach approximates a target distribution, $\pi_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n}$, of the sequence, $x_{1:n}$, of n random variables, that is $\gamma_n(x_{1:n})$ calculates the unnormalized density of $x_{1:n}$.

SIS initilizes each particle $p \in [1, P]$ by sampling from the initial proposal distribution $q_1(x_1^p)$, where x_n^p is the value assigned to the n-th latent variable for particle p. The algorithm then sequentially extends each particle according to the conditional proposal distribution $q_n(x_n^p|x_{1:n}^p)$, where $x_{1:n}^p$ is the sequence of values assigned to the first n latent variables in particle p. After extending a particle p, SIS updates the importance weight $\omega_n^p = \omega_{n-1}^p * \alpha_n(x_{1:n}^p)$. The weight update, defined as

$$\alpha_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1})}, \tag{2}$$

accounts for the discrepancy between the proposal distribution, q_n, and the target distribution, π_n, without normalizing over $x_{1:n}$, which becomes intractable for longer sequences even in discrete domains. The normalizing constant of the target distribution is approximately $Z_n \approx \sum_{p=1}^{P} \omega_n^p$ and the unnormalized density is $\gamma_n(x_{1:n}) \approx \sum_{p=1}^{P} \omega_n^p \text{if} x_{1:n}^p = x_{1:n}$. The particles can also be used to generate an unbiased sample from π_n by choosing a particle p proportional to its weight ω_n^p.

Andrieu et al. [6] shows that to ensure the samples generated by SMC for a Gibbs sampler has the target distribution as the invariant density, the particle filter must be

modified to perform a *conditional SMC update*. This means that the particle filter gaurantees that one of the final particles is assigned the same values as the previous Gibbs iteration. Our implementation of the conditional SMC update reserves one special particle, 0, for which the proposal distribution always chooses the previous iteration's value at that site.

3.1 Sentence Sampling

The `sent` particle filter samples blocks of tag assignments $\mathbf{t}_{1:n}^{S}$ for a sentence, S, composed of tokens, $\mathbf{w}_{1:n}^{S}$. Sampling an entire sentence minimizes the risk of assigning a tag with a high probability given its local context but minimal probability given the entire sentence. Sentences can be sampled by ignoring table counts while sampling a proposal sentence, incorporating them after the fact with a Metropolis-Hastings acceptance test [17]. The Metropolis-Hastings step simplifies the sentence block particle filter further by not requiring the conditional SMC update.

While there is already a tractable dynamic programming approach to sampling an entire sentence based on the Forward-Backward algorithm, particle filtering the sentences PYP-HMM model should prove beneficial. For the trigram HMM defined by the model, the forward-backward sampling approach has time complexity in $O(NT^3)$ for a sentence of length N with T possible tag assignments at each site. Particle filters with P particles can approximate these samples in $O(NTP)$ time, which becomes much faster as T increases.

Sampling of sentence S begins by removing all of the transitions and emissions in S from the table counts, $\mathbf{z}$, resulting in the table counts $\mathbf{z}^{-S}$ of tag assignments $\mathbf{t}^{-S}$ the values assigned to the variables outside of S. For each site index $n \in [1, N]$ in the sentence, the particle filter chooses the new tag assignment, $t_n^{S,p}$, for each particle $p \in [1, P]$ from the sentence proposal distribution,

$$\begin{aligned} q_n^S(t_n^{S,p} | \mathbf{t}_{1:n-1}^{S,p}) \propto\ & P(t_n^{S,p} | t_{n-2}^{S,p}, t_{n-1}^{S,p}, \mathbf{t}^{-S}, \mathbf{z}^{-S}) \\ & \times P(w_n^{S,p} | t_n^{S,p}, \mathbf{t}^{-S}, \mathbf{z}^{-S}, \mathbf{w}^{-S}). \end{aligned}$$

After each new tag is assigned, the particle's weight is updated according to equation (2). The simplicity of the proposal density hints at the advantage of particle filtering over forward-backward sampling: it tracks only P histories and their weights rather than tracking the probability of over all possible histories. Once each particle has assigned a value to each site in the sentence, one tag sequence is chosen proportional to its particle weight, $\omega_N^{S,p}$.

3.2 Type Sampling

The type sampling case for the PYP-HMM is more complicated than the `sent` sampler. The long-range couplings defined by the hierarchical PYP priors strongly influence the joint distribution of tags assigned to tokens of the same word type [5]. Therefore, the affects of the seating decisions of new customers cannot be postponed during filtering as in sentence sampling. To account for this, the `type` particle filter samples sequences

of seating arrangements and tag assignments jointly, $\mathbf{x}^W_{1:n} = (\mathbf{t}^W_{1:n}, \mathbf{z}^W_{1:n})$, for the word-type, W. The final table counts are resampled once a tag assignment has been chosen from the particles.

Tracking the seating arrangement history for each particle adds an additional complication to the `type` particle filter. The exchangeability of seating decisions means that only counts of customers are necessary to represent the history. Each particle represents both a tag sequence, $\mathbf{t}^{W,p}_{1:n}$, and the count deltas, $\mathbf{z}^{W,p}_{1:n}$. The count deltas of each particle are stored in a hash table that maps a dish in one of the CRF restaurants to the number of tables serving that dish and the total number of customers seated at those tables. The count delta hash table ensures that it has sufficient data to calculate the correct probabilities (per equation (1)) by storing any counts that are different from the base counts, $\mathbf{z}^{-W}$, and defering to the base counts for any counts it does not have stored.

At each token occurence n, the next tag assignment, $t^{W,p}_n$ for each particle $p \in [1, P]$ is chosen first according to the word type proposal distribution

$$
\begin{aligned}
q^W_n(t^{W,p}_n | \mathbf{t}^{W,p}_{1:n-1}, \mathbf{z}^{W,p}_{1:n-1}) \propto \\
& P(t^{W,p}_n | c^{-2}_n, c^{-1}_n, \mathbf{t}^{-W,p}_{1:n-1}, \mathbf{z}^{-W,p}_{1:n-1}) \\
& \times P(c^{+1}_n | c^{-1}_n, t^{W,p}_n, \mathbf{t}^{-W,p}_{1:n-1}, \mathbf{z}^{-W,p}_{1:n-1}) \\
& \times P(c^{+2}_n | t^{W,p}_n, c^{+1}_n, \mathbf{t}^{-W,p}_{1:n-1}, \mathbf{z}^{-W,p}_{1:n-1}) \\
& \times P(w^W_n | t^{W,p}_n, \mathbf{t}^{-W,p}_{1:n-1}, \mathbf{z}^{-W,p}_{1:n-1}, \mathbf{w}^{-W,p}_{1:n-1}).
\end{aligned}
$$

In this case, $c^{\pm k}_n$ represents a tag in the context of site t^W_n offset by o, while $\mathbf{t}^{-W,p}_{1:n-1}$, $\mathbf{z}^{W,p}_{1:n-1}$, and $\mathbf{w}^{-W,p}_{1:n-1}$ represent the tag assignments, table counts, and word token values chosen by particle p as well as the values at all of the sites where a word token of type W does not appear. This proposal distribution ignores changes to the seating arrangement between the three transitions involving the site n. The specific seating arrangement of a particle is chosen after the tag choice, at which point the weights are updated by the result of equation (2). As with the `sent` sampler, once all of the particles have been sampled, one of them is sampled with probability proportional to its weight. This final sample is a sample from the true target probability.

As mentioned earlier, the sequence of particle approximations do not have the target distribution as invariant unless they use the conditional SMC update. Therefore, a special 0^{th} particle is automatically assigned the value from the prior iteration of the Gibbs sampler at each site n, though the proposal probability $q^W_n(t^{W,0}_n | \mathbf{t}^{W,p}_{1:n-1}, \mathbf{z}^{W,p}_{1:n-1})$ still has to be calculated to update the weight $\omega^{W,p}_n$ properly. This ensures that the `type` sampler has a non-zero probability of reverting to the prior iteration's sequence.

4 Experiments and Results

We aim to explore several aspects of both the PG sampler and the specifics of the PoS inference task. Firstly, the motivation of the PG implementations is to allow better inference on tightly coupled models. This suggests that the PG samplers should mix better than the local sampler, finding the mode of the model in fewer iterations. The type sampler should perform especially better with the character-LM, which assigns a higher

likelihood when words of the same type are labeled with the same tag. If the PG approach does find more likely modes, does that correspond to gains in practical measures like accuracy? Finally, how does the number of particles in PG samplers influence the inference? We hypothesize that increasing the number of particles decreases the variance of the posterior likelihoods as the chance of generating a population dominated by likely modes increases. However, the marginal improvement from additional particles should exhibit diminishing returns.

Section 4.1 describes the corpora on which these hypotheses are tested. We evaluate the performance of the samplers with two approaches. The first approach is an analysis of the samplers as inference algorithms. Each approach should tend toward a mode in the distribution as it mixes, resulting in more likely restaurant configurations. Section 4.2 analyzes the particle filter based samplers with various numbers of particles in an effort to understand how they behave. Then, section 4.3 evaluates each of the proposed approaches on PoS inference tasks from several languages. These results allow a practical comparison with other PoS inference approaches.

4.1 Data

Section 4.2 tests and compares several different approaches with a number of parameters. To simplify the procedures, all of the tests in the analysis are performed on a reduced version of the Penn Wallstreet Journal (WSJ) treebank, similar to Gao and Johnson [17] and Goldwater and Griffiths [18]. This reduced corpus is composed of the first 10,000 sentences in the Penn WSJ treebank, with 240,236 tokens. Additionally, this corpus uses the same, 17 tag, reduced tagset as Goldwater and Griffiths [18], developed by Smith and Eisner [19].

Section 4.3 compares the many-to-one (M-1) accuracy of the induced tag assignments on the full Penn WSJ treebank corpus, as well as the corpora from the CoNLL-X shared language task [20]. M-1 accuracy assigns the induced syntactic categories to the PoS of the most tokens of that category.

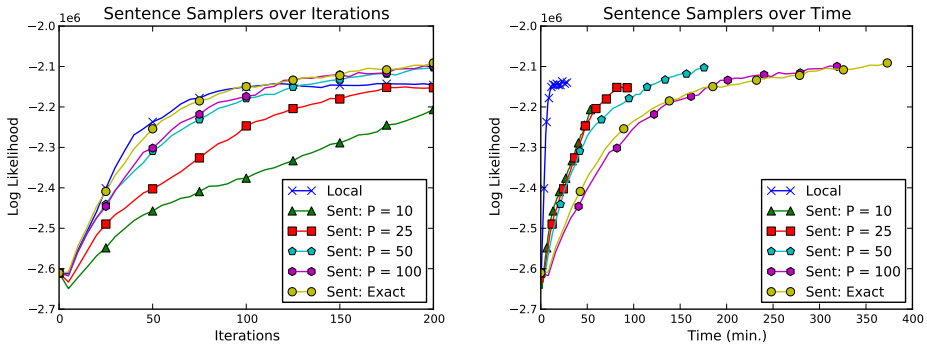
4.2 SMC Analysis

Before comparing the performance of the PG samplers to other inference methods, we wish to learn more about the approaches themselves. It is not obvious how well the benefits of block sampling transfer to SMC based approaches. Both the `sent` and `type` samplers are novel approaches to computational linguistics, and many of their properties are unclear. For example, the samples generated from the particle filter should have a higher variance than the target distribution. If the variance is too high, the sampler will be slower to converge. While additional particles lower the relative variance, they also increase the run time linearly. We hypothesize that there is a threshold of particles necessary to ensure that some are high likelihood sequences, beyond which inference gains are minimal the additional computational expense is wasted.

Like other MCMC methods, particle Gibbs generates a sequence of samples from the posterior distribution of the model in question. The PG sampler should mix more quickly than a local sampler because it takes larger steps. These larger steps should move the PG sampler to a more likely mode in fewer iterations. The results in this

section measure the power of the inference as the rate at which the posterior likelihood of the restaurant configuration increases.

The sentence based sampler, `sent`, samples from a distribution that can be exactly computed, facilitating comparisons between the exact sampler and the SMC approach. Figure 1 compares the posterior log-likelihoods of the `sent` sampler and the exact sentence sampler over 200 iterations. As expected, the likelihoods of the particle filters approach that of the exact sentence sampler as the number of particles increases from 10 to 100.



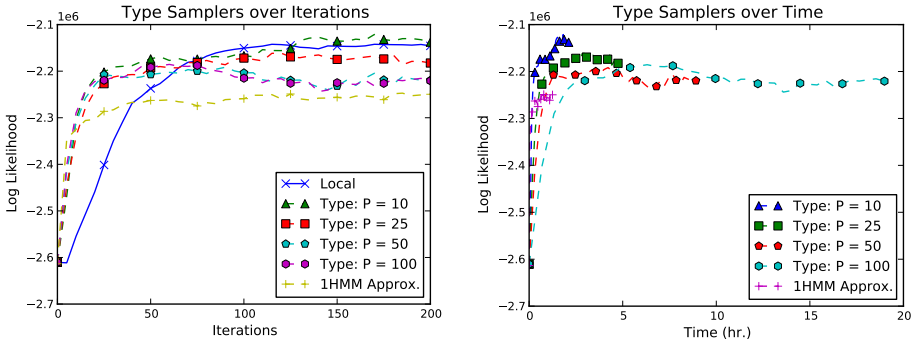
(a) Iterations (b) Time (min.)

Fig. 1: Posterior Log-Likelihood of PYP-HMM inference over iterations (a) and time (b) with exact as well as PG `sent` sampler with various numbers of particles. Except for the local sampler, which ran for 300 iterations in the time graph, the samplers were run for 200 iterations each, the end of a line represents the time to finish those 200 iterations.

Figure 2 compares the table configuration log-likelihood of the 1HMM approximation implemented by Blunsom and Cohn [4] with the `type` particle filter based sampler as well as the local sampler and the exact block sentence sampler. Unlike the sentence based block sampler, `type` sampler cannot be exactly calculated, even with the 1HMM approach of constraining inference to only consider sequences that assign the same tag to every token of the same word type. The 1HMM sampler approximates these probabilities using expected table counts. Theoretically, the `type` sampler should be a better approximation, being gauranteed to approach the true distribution as the number of particles increases. The results suggest that the `type` sampler can perform better than the 1HMM sampler with few particles. Unlike the `sent` sampler, the `type` sampler performs well even with fewer particles than the number of PoS categories in the tagset. These results suggest that the choice of block has a strong influence on the resulting sampler.

Interestingly, the local sampler reaches a higher mode than the 1HMM approach before the fiftieth iteration. However, earlier work by Blunsom and Cohn [4] found that

the 1HMM approximation achieved a consistently higher M-1 accuracy than the local sampler. Section 4.3 confirms the same result. This reveals a disconnect between the likelihood under the PYP-HMM model and the M-1 accuracy. Additionally, both the `type` and 1HMM samplers reach likely modes within 30 iterations, after which they plateau and the additional cost of these approaches is wasted.



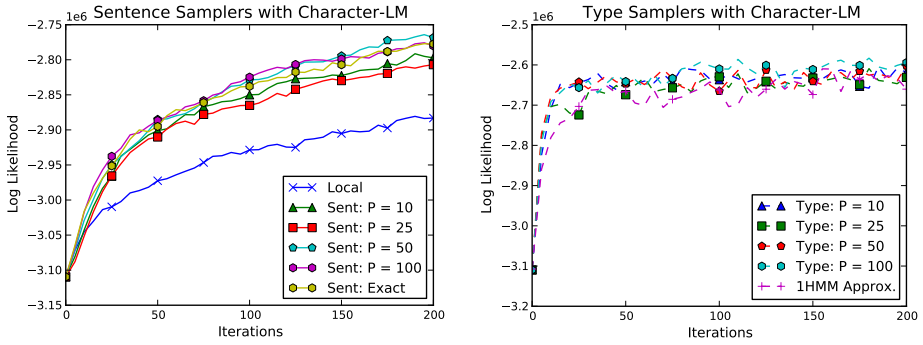
(a) Iterations (b) Time (hr.)

Fig. 2: Posterior Log-Likelihood of PYP-HMM inference over iterations (a) and time (b) with the `type` sampler as well as the 1HMM approximation proposed by Blunsom and Cohn [4] with various numbers of particles. Except for the local sampler, which ran for 300 iterations in the time graph, the samplers were run for 200 iterations each, the end of a line represents the time to finish those 200 iterations.

An interesting variant of the basic PYP-HMM model replaces the uniform base of the emission distribution with a bigram character language model (PYP-HMM-LM) [4]. In addition to allowing the PYP-HMM to recognize basic word morphology, the character language model creates a strong bias toward only one tag per word type. Figure 3a shows that the PYP-HMM-LM model is too tightly coupled for either the local or the `sent` samplers to mix quickly as they did with the simpler PYP-HMM model. The PYP-HMM-LM model reveals a weakness of the simple local Gibbs sampler approach, because the probability of a word being emitted by a new state is so low that sampler is highly likely to choose the same PoS tag assignment as other occurences of that word. The additional context of the `sent` sampler results in bigger steps and faster mixing than the local sampler, but most of the variables influencing probability of and particular tag assignment are in different sentences.

On the other hand, the `type` and 1HMM samplers sample all of the tags assignments for words of the same type simultaneously. Figure 3b shows that the result is drastically faster mixing. The `type` sampler is just as strongly influenced to assign all words of the same type to the same PoS category, but there are no same-word-type assignments to bias the choice of which one tag they will all get. This strong correlation between the tags assigned to words of the same type may explain the `type` sampler's

strong performance with so many fewer particles than necessary for the `sent` sampler: once a tag has been assigned to a sufficient number of words of the same type, that same tag will be progressively more likely to be chosen again. Perhaps a one-tag-per-type particle filter can take advantage of this fact to simplify the proposal distribution.



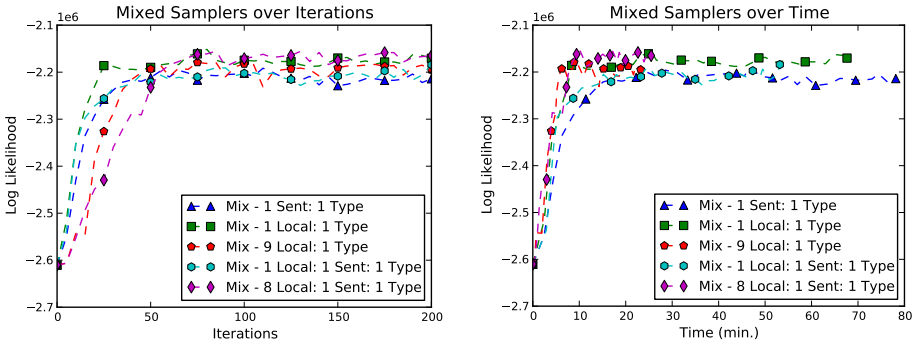
(a) `sent` (b) `type`

Fig. 3: Posterior Log-Likelihood of PYP-HMM-LM inference with the `sent` sampler (a), and the `type` sampler as well as the 1HMM approximation proposed by Blunsom and Cohn [4] (b) with various numbers of particles.

The fact that the `type` sampler requires so few particles for inference relative to the `sent` sampler suggests that the choice of block can heavily influence the performance of a block Particle Gibbs sampler. The optimal block for a given model is not obvious and may be difficult to determine empirically. On the other hand, the large and expensive steps taken by a Particle Gibbs sampler are likely only necessary when faster methods reach modes from which they are slow to escape. A mixed sampler that only occasionally takes large steps with Particle Gibbs might achieve similar results with less computational cost. Such an approach need not even rely on any specific block partitioning, particle filters do not place many restrictions on the distributions from which they generate samples.

There are a multitude of possible combinations of mixed samplers, figure 4 demonstrates the performance of a few instantiations of one simple approach. Each of the mixed samplers in figure 4 randomly choses on of either the local, `sent`, or `type` samplers (the numbers in the legend describe the ratio of each sampler in the mixture.) For the sake of simplicity, each mixed sampler uses the same number of particles for the `sent` and `type` samplers, the mixed samplers in figure 4 all use ten particles. While there is a high degree of variance, the mixed samplers all show the rapid migration toward the mode shown by the `type sampler`, but each iteration takes less time on average because of the `sent` and local samplers. The results in figure 4b suggest that the computational costs of the Particle Gibbs samplers can be mitigated by mixed sampling without eliminating the benefits from particle filtering. Finally, the samplers with

the `sent` sampler in the mix performed quite well with ten particles despite the its poor performance with the same number of particles on its own. Perhaps mixing the block samplers diminishes the particle cost of poor block choice.



(a) Iterations (b) Time (min.)

Fig. 4: Posterior Log-Likelihood of PYP-HMM inference with various mixed samplers over 200 iterations ((a)) and over time ((b)). The colon separated lists provide the ratio of each mixed sampler: the sampler run on any given iteration is proportional to the number next to it. In each sampler, the particle filters are run using 10 particles. The samplers were run for 200 iterations each, the end of a line represents the time to finish those 200 iterations.

4.3 Unsupervised Part-of-Speech Tagging

Table 1 compares the M-1 accuracy of the `sent` and `type` particle filter samplers, from sections 3.1 and 3.2, with 100 particles each. Each site in a corpus is assigned the most commonly visited tag assignment at that site over all iterations. The particle filter based samplers rarely score a higher accuracy than even the local sampler, which completes 500 iterations before the particle filters complete 200.

While figures 1a and 2a show that all of the samplers surpass the 1HMM sampler in likelihood, the accuracies of the 1HMM and 1HMM-LM approximations remain well above the other approaches. This suggests that there are high-likelihood assignments that produce lower accuracy results, presumably related to the fact that the `type` sampler is not restricted to assignments with exactly one tag for each word type. If the model assigns equal likelihood to these assignments, inference will not be able to distinguish between them.

Excepting Arabic, the Type-10-LM sampler outperforms all of the other non-restrictive approaches, suggesting that the PYP-HMM-LM model may be a more accurate representation of PoS. As noted in section 4.2, the PYP-HMM-LM model is biased toward

Table 1: Many-to-1 accuracies on CoNLL and Penn-Treebank Wall Street Journal corpora for sentence- (Sent) and type- (Type) based filtering. The table lists the average M-1 accuracy measured according to the maximum marginal tag assignments over 3 seperate runs after 200 iterations for the `sent`, `type`, 1HMM and 1HMM-LM samplers, and 500 iterations for the HMM local sampler.

Language	Sent-100	Type-100	Type-10-LM	Local	1HMM	1HMM-LM	Tokens	Tag types
WSJ	69.8%	70.1%	74.7%	70.2%	75.6%	77.5%	1,173,766	45
Arabic	53.5%	57.6%	51.3%	56.2%	61.9%	62.0%	54,379	20
Bulgarian	64.8%	67.8%	71.6%	67.6%	71.4%	76.2%	190,217	54
Czech	59.8%	61.6%	65.2%	64.5%	65.4%	67.9%	1,249,408	12^c
Danish	65.0%	70.3%	74.9%	69.1%	70.6%	74.6%	94,386	25
Dutch	61.6%	71.6%	70.1%	64.1%	73.2%	72.9%	195,069	13^c
Hungarian	61.8%	61.8%	68.5%	64.8%	69.6%	73.2%	131,799	43
Portuguese	59.4%	71.1%	74.4%	68.1%	72.0%	77.1%	206,678	22
Spanish	66.3%	69.1%	75.3%	68.5%	74.7%	78.8%	89,334	47
Swedish	62.9%	63.5%	68.3%	67.6%	67.2%	68.6%	191,467	41

fewer tags per type than the more standard HMM model, resulting in an average number of tags per word type that is closer to the true value. Even so, figure 3b shows that the `type` sampler mixes at least as well as the 1HMM sampler with the character language model, yet the 1HMM sampler still scored a higher many-to-one accuracy on all languages other than Danish. This discrepancy between model likelihood and accuracy suggests that a different model is necessary to further improve unsupervised PoS induction.

5 Conclusion

This paper presented a novel application of the Particle Gibbs sampler approach to the computational linguistic inference application of unsupervised PoS induction. Such approaches show great potential for inference, especially in highly dependent distributions e.g. non-parametric Bayesian applications. While this power generally comes and the expense of significantly increased computation, results show that a mixed sampler that only occasionally performs a Particle Gibbs sampling step can achieve similar results in a fraction of the time. Additionally, the type particle filter itself can be largely run in parallel, only bottlenecking when the particle weights need to be normalized. Further expansion of the basic ideas presented will enable scalable inference in otherwise intractable models.

References

1. Goldwater, S., Griffiths, T., Johnson, M.: Interpolating between types and tokens by estimating power-law generators. In Weiss, Y., Schölkopf, B., Platt, J., eds.: Advances in Neural Information Processing Systems 18. MIT Press, Cambridge, MA (2006) 459–466

2. Liang, P., Petrov, S., Jordan, M., Klein, D.: The infinite PCFG using hierarchical Dirichlet processes. In: Proc. of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-2007), Prague, Czech Republic (2007) 688–697
3. Cohn, T., Goldwater, S., Blunsom, P.: Inducing compact but accurate tree-substitution grammars. In: HLT-NAACL '09: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (2009) 548–556
4. Blunsom, P., Cohn, T.: A hierarchical Pitman-Yor process hmm for unsupervised part of speech induction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, USA, Association for Computational Linguistics (June 2011) 865–874
5. Liang, P., Jordan, M.I., Klein, D.: Type-based MCMC. In: North American Association for Computational Linguistics (NAACL). (2010)
6. Andrieu, C., Doucet, A., Holenstein, R.: Particle markov chain monte carlo methods. Journal Of The Royal Statistical Society Series B **72**(3) (2010) 269–342
7. Canini, K.R., Shi, L., Griffiths, T.L.: Online inference of topics with latent Dirichlet allocation. In van Dyk, D., Welling, M., eds.: Proceeings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS*09). (2009) 65–72
8. Borschinger, B., Johnson, M.: A particle filter algorithm for bayesian wordsegmentation. In: Proceedings of the Australasian Language Technology Association Workshop 2011, Canberra, Australia (December 2011) 10–18
9. Teh, Y.W.: A hierarchical bayesian language model based on Pitman-Yor processes. In: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. ACL-44, Morristown, NJ, USA, Association for Computational Linguistics (2006) 985–992
10. Gordon, N.J., Salmond, D.J., Smith, A.F.M.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. Radar and Signal Processing, IEE Proceedings F **140**(2) (April 1993) 107–113
11. Jasra, A., Doucet, A., Stephens, D., Holmes, C.: Interacting sequential Monte Carlo samplers for trans-dimensional simulation. Computational Statistics & Data Analysis **52**(4) (January 2008) 1765–1791
12. Beaumont, M.A.: Estimation of Population Growth or Decline in Genetically Monitored Populations. Genetics **164**(3) (July 2003) 1139–1160
13. Fernandez-Villaverde, J., Rubio-Ramirez, J.F.: Estimating macroeconomic models: A likelihood approach. Review of Economic Studies **74**(4) (October 2007) 1059–1087
14. Kitagawa, G.: Monte carlo filter and smoother for non-gaussian nonlinear state space models. Journal Of Computational And Graphical Statistics **5**(1) (1996) 1–25
15. Fearnhead, P., Wyncoll, D., Tawn, J.: A sequential smoothing algorithm with linear computational cost. Technical report, Department of Mathematics and Statistics, Lancaster University (2008)
16. Doucet, A., Johansen, A.M. In: A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later. Oxford University Press (2009)
17. Gao, J., Johnson, M.: A comparison of bayesian estimators for unsupervised hidden markov model pos taggers. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. EMNLP '08, Morristown, NJ, USA, Association for Computational Linguistics (2008) 344–352
18. Goldwater, S., Griffiths, T.: A fully bayesian approach to unsupervised part-of-speech tagging. In: Proc. of the 45th Annual Meeting of the ACL (ACL-2007), Prague, Czech Republic (June 2007) 744–751

19. Smith, N.A., Eisner, J.: Contrastive estimation: Training log-linear models on unlabeled data. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL), Ann Arbor, Michigan (June 2005) 354–362
20. Buchholz, S., Marsi, E.: Conll-x shared task on multilingual dependency parsing. In: Proceedings of the Tenth Conference on Computational Natural Language Learning. CoNLL-X '06, Morristown, NJ, USA, Association for Computational Linguistics (2006) 149–164