Fast Search for Infinite Latent Feature Models

Piyush Rai and Hal Daumé III
School of Computing, University of Utah
{piyush,hal}@cs.utah.edu

Abstract

We propose several search based alternatives for inference in the Indian Buffet Process (IBP) based models. We consider the case when we only want a maximum a posteriori (MAP) estimate of the latent feature assignment matrix. If true posterior samples are required, these MAP estimates can also serve as intelligent initializers for MCMC based algorithms. Another advantage of the proposed methods is that they can process one observation at a time making it possible to do inference in an online setting. Experimental evidences suggest that these algorithms can give us computational benefits of an order of magnitude over Gibbs sampling (or its sequential variant - the particle filter) traditionally used in IBP based models.

1 Introduction

The Indian Buffet Process (IBP) [3] is a flexible nonparametric Bayesian model used to discover the latent feature representations of a set of observations. Given an $N \times D$ matrix X of N observations having D dimensions each, the goal is to represent X as $ZA+E$. Here Z is an $N \times K$ binary matrix denoting which latent features are present in each observation, A is a $K \times D$ matrix consisting of feature scores, and E consists of observation specific noise. The flexibility of IBP comes from the fact that one does not need to ***a priori*** specify K - the number of latent features. IBP defines a prior on the binary matrix Z such that it can have a potentially unbounded number of columns (and thus the number of latent features), limited only by the number of observations. The IBP has a nice culinary analogy of N customers coming to an Indian buffet and making selections from an infinite array of dishes. The resulting customer-dish pattern is reflected in the binary feature assignment matrix Z.

Inference in the IBP based models has typically been sampling based. Sampling based approaches proposed for IBP include vanilla Gibbs sampling [3], and slice sampling based on the stick-breaking representation of the IBP [6]. Another more efficient alternative is to use particle filters [5]. Recently, [2] proposed a variational approximation to the posterior based on the truncated stick-breaking approximation of the IBP. However, these techniques have their own shortcomings. Sampling based schemes are guaranteed to give the exact solution in the limit, but in practice they often exhibit slow mixing. Variational methods, although faster than sampling based methods, can be difficult to design and implement. As is usually the case, often we require only the MAP sample from the set of samples obtained from the posterior, discarding all other samples. This naturally leads to the following question: *If all we care about is a single MAP assignment, why not just find one directly?* In this paper, we propose search algorithms such as A^* [4] and beam search as viable alternatives to the above techniques for cases where only a single MAP assignment of the Z matrix is desired.

Finding exact MAP estimate for the binary matrix Z is NP-hard (can be shown via graph-partitioning). Our proposed search based algorithms can find *exact* MAP estimates for small problems, and *approximate* MAP estimates for large problems. In addition, if samples from the true posterior are desired then the search based MAP estimates can serve as good initializers for MCMC, resulting in faster convergence.

2 Search based MAP Estimate for IBP

Our beam-search algorithm for IBP takes as input the set of observations, a scoring function, and a maximum beam size. The algorithm processes one observation at a time and maintains a max-queue of the *partial* latent feature assignment matrices (i.e., of observations processed so far). The scoring function is used to rank candidate matrices on the queue and maximum beam size specifies the maximum number of candidates allowed on the queue at any time. In each iteration, the most promising candidate $\mathbf{Z}^0$ is removed from the queue, and is expanded with the set of all possible feature assignments for the next i^{th} observation. For the possible expansions, we consider 2^K possibilities for assigning the existing dishes, and additionally 0 and $\max\{1, \lfloor \alpha/i \rfloor\}$ new dishes for each of these cases (note: $\lfloor \alpha/i \rfloor$ is the mode of the number of new dishes chosen by the i^{th} customer in the IBP culinary analogy). Scores are computed for each of the new candidates and these are placed in the queue. If the beam size is not infinite then we also drop the lowest scoring elements so as to maintain the maximum queue size. We stop at the point when the number of rows in the matrix Z removed from the queue equals the total number of observations.

The search algorithm is guaranteed to find the MAP feature assignment matrix if the beam size is ∞ and the scoring function g is *admissible*. Being admissible means that it should *over-estimate* the posterior probability of *best possible* feature assignment $\mathbf{Z}$ that agrees with $\mathbf{Z}^0$ on the first N^0 observations (i.e., have *identical* first N^0 rows). Denoting this condition as $\mathbf{Z}|N^0 = \mathbf{Z}^0$, admissibility can be written formally as $g(\mathbf{Z}^0, \mathbf{X}) \geq \max_{\mathbf{Z}|N^0=\mathbf{Z}^0} p(\mathbf{Z}, \mathbf{X})$, with the equality holding for $N^0 = N$.

The admissible scoring functions provably lead to optimal MAP estimates. However, the NP-hardness of the MAP problem implies that these can be inefficient. For efficiency reasons, therefore, it is often useful to have scoring functions that occasionally *under-estimate* the true posterior probability. These functions are no longer guaranteed to find the optimal solution but can be more efficient since g can be tighter, even if it is not a strictly upper bound.

The posterior probability $\mathbf{Z}$ given $\mathbf{X}$, $p(\mathbf{Z}|\mathbf{X})$ is proportional to $p(\mathbf{Z}, \mathbf{X})$ which can be again factored as $p(\mathbf{Z})p(\mathbf{X}|\mathbf{Z})$. The probability $p(\mathbf{Z})$ of feature assignment matrix is given by $P(\mathbf{Z}) = \frac{\alpha^K}{\prod_{h=1}^{2^N-1}} exp(-\alpha H_N) \prod_{k=1}^{K} \frac{(N-m_k)!(m_k-1)!}{N!}$, where H_N is the N^{th} harmonic number, and $m_k = \sum_i Z_{ik}$. The likelihood term $p(\mathbf{X}|\mathbf{Z})$ is Gaussian in the linear Gaussian model we consider in this paper.

An upper bound on the posterior can be obtained by independently upper-bounding the prior probability $p(\mathbf{Z})$ and the likelihood $p(\mathbf{X}|\mathbf{Z})$. In fact, as we shall show, it is possible to even explicitly maximize the prior. Unfortunately, the same is not true for the likelihood term, and we therefore propose several heuristics for this maximization.

Our search algorithm is akin to the A^* search where we optimize a *path-cost-so-far* function plus a *cost-to-goal* function. In our case, we rank a candidate feature assignment matrix by computing its score that is a summation of the posterior probability upto first N^0 observation, and the upper bound on the posterior probability corresponding to the remaining observations. In keeping with the culinary metaphor of IBP, in the rest of the exposition, we denote observations by customers, and features by dishes.

3 Maximizing the Prior

Given the customer-dish assignments for the first N^0 customers, it is possible to explicitly compute the dish assignments for the remaining customers that maximizes the probability $P(\mathbf{Z})$. For this maximization, we need to consider two cases for the remaining customers: (a) sampling the already selected dishes, and (b) sampling the new dishes.

Sampling the already selected dishes: Given an $N^0 \times K$ matrix $\mathbf{Z}^0$ for first N^0 customers, the $(N^0+1)^{th}$ customer chooses an already selected dish k only if it has been chosen previously by more than half the customers (i.e., the *majority*). Now, from the perspective of the prior $P(\mathbf{Z})$, whether the subsequent customers choose this dish or not depends on the choice made by the $(N^0+1)^{th}$ customer. If this customer chose the k^{th} dish, then all remaining customer will choose it since a majority of previous customers would have chosen this dish. If however the the k^{th} dish is skipped by this customer, then none of the remaining customers either will choose this dish since it was skipped by a majority of previous customers. For dish k, the score is given by $[p_k^{x_k}(1-p_k)^{(1-x_k)}]^{(N-N^0)}$, a product of $(N-N^0)$ binomials, each with the same parameters p_k and x_k. Here $x_k = (m_k \geq N^0/2)$, $p_k = (m_k + N - N0)/N$, and m_k is the number of previous customers who chose the k^{th} dish. The total score for this part is given by the sum of individual scores for each of the existing dishes.

Sampling the new dishes: In the IBP culinary metaphor, the i^{th} customer selects $Poisson(\alpha/i)$ number of new dishes so the prior would be maximized if customer i selects a number of dishes equal to the *mode* of this number which is $\lfloor \alpha/i \rfloor$. The score contribution of this part for $P(\mathbf{Z})$ is given by:

$$\prod_{n=N^0+1:N} \frac{(\alpha/n)^{\lfloor \alpha/i \rfloor !} \mathcal{E}xp(-\alpha/n)}{\lfloor \alpha/i \rfloor !}$$

The product involving the $\mathcal{E}xp$ term just requires a harmonic mean of $(N-N^0)$ numbers. For the terms involving $\lfloor \alpha/i \rfloor$, we only need to care about those for which $\lfloor \alpha/i \rfloor > 0$. Therefore, this computation is inexpensive.

4 Maximizing the Likelihood

Unlike the prior term, an explicit maximization is not possible for the likelihood term because a future customer would not have been assigned dishes yet, precluding the corresponding likelihood computation. We therefore need heuristics to do this maximization.

Trivial upper-bound for the Likelihood: Given the matrix $\boldsymbol{Z}^0$ having N^0 many rows, a possible trivial upper bound $P(\boldsymbol{X}|\boldsymbol{Z})$ can be obtained by only considering likelihood over the first N^0 customers. This is an admissible heuristic function since it gives a probability one to each of likelihood terms for the remaining customers, and would therefore would always overestimate the upper-bound on the true likelihood. This function is given by $g_{Trivial}(\boldsymbol{X} \mid \boldsymbol{Z}^0) = P(\boldsymbol{X}_{1:N^0} \mid \boldsymbol{Z}^0)$.

Computing the marginal likelihood term on the right hand side requires all the N^0 observations seen thus far, and thus has a quadratic complexity which can be undesirable. We therefore use the uncollapsed likelihood which factorizes over the observations as $p(\mathbf{X}|\mathbf{Z}, \mathbf{A}) = \prod_n p(\mathbf{X}_n|\mathbf{Z}_n, \mathbf{A})$. To do this, we need to maintain the posterior on $\mathbf{A}$ and update it sequentially. Rank-1 updates can be used to achieve this. As an analogy to standard A^* search, using this trivial cost function is essentially equivalent to using a path cost with zero heuristic function. Such a scoring function will be expected to lead to an inefficient search.

Other heuristics for likelihood maximization: The trivial function discussed above is admissible but it is a loose upper-bound since it does not take into account the dish assignments of any of the future customers, and would therefore be inefficient. To do so, we must find some way of accounting for the dish selection for the remaining customers. Here we discuss some alternatives.

One possibility is to use a function which is significantly tighter, much more efficient, but no longer admissible and therefore the search is not guaranteed to find the global optimal solution. This *inadmissible* function is given by $g_{Inad}(\boldsymbol{X} \mid \boldsymbol{Z}^0) = g_{Trivial}(\boldsymbol{X} \mid \boldsymbol{Z}^0) \prod_{n=N^0+1}^{N} P(\boldsymbol{X}_n|\boldsymbol{Z}_n)$, where $\boldsymbol{Z}_n$ consists of all zeros up to column K, followed by a one. It means that we are assigning a new dish to each new customer. This is an inadmissible heuristics since it is always preferable to assign the same set of dishes to two customers if both are identical. When using this heuristic, it is important to present the data points in an order that does not hurt the heuristic too much. So we sort all the datapoints by increasing marginal likelihood and present them in that order. This is a reasonable thing to do since data points with high marginal likelihood are more likely to select their own new dishes.

Another way to incorporate the dish assignment of future customers in the likelihood term is to first do a *coarse level* of feature assignment. Given the set of observations $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$, we first cluster them using some fast clustering algorithm (we use an online clustering algorithm from [1]) to do this. Having obtained a clustered representation of the data, we pick one representative point from each cluster and run the IBP search algorithm (using the trivial scoring function described above) on these cluster representative observations. This gives us a coarse feature assignment for the representative points. We then run the IBP search on the entire set of observations and, while computing the likelihood (heuristic) of a future customer n, we assign it the same set of dishes as assigned to the representative of the cluster the n^{th} observation belongs to.

5 Experiments

We have compared our search based algorithms against vanilla Gibbs sampler for the IBP on several synthetic datasets, 3 of which are reported in table-1. Two of the synthetic datasets used here were generated by drawing the matrix Z from the IBP prior and then generating the observations. The third dataset is a subset of the image block dataset from [5]. The results are shown in table-1. As the results suggest, the search based approaches yields results comparable to Gibbs sampling and is an order of magnitude faster. The search algorithms also discovered the correct number of latent features for datasets with known ground-truth. In the table, the algorithm Search-1 uses the admissible heuristic and algorithm Search-2 uses the inadmissible heuristic. Small beam sizes ($\sim$5-20) were used in all the experiments and the search algorithms used fixed hyperparameters (α and noise hyperparameters in the linear-Gaussian model). The MSE (mean squared error) score computation requres the feature score matrix A. Since our search algorithm only finds Z, we generate A by sampling it from its posterior given Z. logP denotes logP(X,Z). The Gibbs samplers were run for just as many iterations as were typically needed to achieve convergence on these datasets.

	Dataset-1 (N=20, D=16)			**Dataset-2** (N=100, D=100)			**Dataset-3** (N=200, D=16)		
	Time	MSE	logP	Time	MSE	logP	Time	MSE	logP
Gibbs	6.5869	0.1012	-129.6	27.3176	0.0087	-4734.2	55.63	0.0041	-3859.6
Search-1	1.0153	0.0094	-127.4	7.1564	0.0098	-4711.6	17.95	0.0044	-4276.7
Search-2	1.0149	0.0092	-127.8	5.6550	0.0110	-4932.1	19.12	0.0047	-4366.4

In terms of scalability, as the number of observations grow, our search based algorithms on most datasets scale almost linearly which is much better than Gibbs sampling. Besides this, from the perspective of online inference, our early results have also demonstrated better scalability than particle filters [5] as the number of observations grows (not reported here). Finally, we note here that the search based approach yields MAP estimate of the latent feature assignment matrix. However, if true posterior is desired then we have found empirically that the MAP estimates can serve as good initializers for the vanilla Gibbs sampler, resulting in higher post-convergence log-probabilities (in addition to quicker convergence).

6 Discussion and Future Work

It is possible to speed up the search algorithms proposed here even further. Note that when a candidate is removed from the queue and expanded with the possible feature assignments for the next observation, we need to consider all 2^K possible candidates, compute their scores, and place them on the queue. This can be expensive for cases where K is expected to be large. An alternative to this would be to do the beam search by expanding along the *columns* of the Z matrix for a given row, considering one dish at a time, which would assuage the exponential dependence on K. Besides, heuristics used for likelihood maximization are critical to getting tighter bounds for the posterior and it would be interesting to consider other possible heuristics that result in tighter bounds, and evaluate them on a wider range of real-world datasets. Furthermore, in this paper we consider the linear-Gaussian model for the data. It would be interesting to apply the search based approach to other models.

References

[1] Hal Daumé III. Fast search for dirichlet process mixture models. In *AIStats*, 2007.

[2] F. Doshi-Velez, K. T. Miller, J. V. Gael, and Y. W. Teh. Variational inference for the indian buffet process. In *AIStats*, 2009.

[3] Z. Ghahramani, T. L. Griffiths, and P. Sollich. Bayesian Nonparametric Latent Feature Models. In *Bayesian Statistics 8*, 2007.

[4] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. In *IEEE Transactions on Systems Science and Cybernetics*, 1968.

[5] F. Wood and T. L. Griffiths. Particle filtering for nonparametric bayesian matrix factorization. In *NIPS*, 2007.

[6] Z Ghahramani Y. W. Teh, D. Görür. Stick-breaking construction for the indian buffet process. In *AIStats*, 2007.