Posterior sampling over partitioned data with Stein variational gradient descent.

Feynman Liang, Qiang Liu, Michael Mahoney

September 12, 2018

1 Introduction

Drawing samples from a posterior distribution $P(\theta \mid \mathcal{D})$ is a fundamental tool powering many applications of Bayesian statistics. Stein variational gradient descent (SVGD) [2] is a recently developed particle-based algorithm for performing posterior sampling from arbitrary distributions.

Modern datasets are increasingly large, and in many problems of interest the data X does not fit on a single machine. In such environments, development of distributed algorithms which require only subsets of the data at each iteration are of particular interest. Furthermore, the increased communication costs incurred from network I/O between machines motivates the need for communication efficient algorithms.

In this work, we propose a new extension to SVGD which can be applied when data and/or particles are partitioned across multiple machines. Our work closes the scalability gap for SVGD and enables it to scale to big datasets and a large number of potentially high dimensional particles.

2 Background

Given some probability measure ν with density p (e.g. the posterior distribution $P(\theta \; mid \mathcal{D})$), we want to find a particle approximation $\{\theta_j\}_{j=1}^n$ whose empirical measure $\hat{\mu}_n = \frac{1}{n}\sum_{j=1}^n \delta_{\theta_j}$ converges weakly

$$\mu_n \Rightarrow \nu \tag{1}$$

SVGD does this by initialiing $\theta_j \overset{\sim}{\text{iid}} \mu$ and performs iterative updates

$$T_{\epsilon,\phi}(\theta) = \theta + \epsilon\phi(x) \tag{2}$$

The choice of ϕ should make the approximation better, and is formalized by the functional optimization

$$\max_{\phi \in \mathcal{H}} \left\{ -\frac{d}{d\epsilon} KL(T\mu \mid \nu) \mid_{\epsilon=0} \colon \|\phi\|_{\mathcal{H}} \leq 1 \right\} \tag{3}$$

where $\mathcal{H}$ is a Banach space.

[2] made the connection between this objective and the Stein operator from statistics

$$-\frac{d}{d\epsilon}KL(T\mu \mid \nu) \mid_{\epsilon=0} = \mathbb{E}_\mu \mathcal{S}_p \phi \quad \text{where} \quad \mathcal{S}_p\phi(x) = \nabla \log p(x)^\intercal \phi(x) + \nabla \cdot \phi(x) \tag{4}$$

The optimization thus is equivalent to

$$\mathbb{D}(\mu \mid \nu) = \max_{\phi \in \mathcal{H}} \{\mathbb{E}_\mu \mathcal{S}_p \phi : \|\phi\|_{\mathcal{H}} \leq 1\} \tag{5}$$

which is called the *Stein discrepancy*. Restricting $\mathcal{H}$ to a RKHS renames this quantity to the *Kernelized Stein discrepancy*.

[1] showed that the optimal solution

$$\phi^*_{\mu,p}(\cdot) \propto \mathbb{E}_{x\sim\mu}\left[\nabla \log p(x) k(x,\cdot) + \nabla_x k(x,\cdot)\right] \tag{6}$$

suggesting an iterative algorith given in algorithm 1.

Algorithm 1 The SVGD algorithm

Input: Likelihood $p(\mathcal{D} \mid \theta)$, prior $p(\theta)$, and initial particles $\{\theta_i^0\}_{i=1}^n$
Output: Particles $\{\theta_i\}_{i=1}^n$ approximating the posterior $p(\theta \mid \mathcal{D})$
  for iteration ℓ **do**
    for particle $i = 1$ **to** n **do**
      $\theta_i^{\ell+1} \leftarrow \theta_i^\ell + \epsilon_\ell \hat{\phi}^*(\theta_i^\ell)$ where $\hat{\phi}^*(\theta) = \frac{1}{n}\sum_{j=1}^n \left[k(\theta_j^\ell, \theta)\nabla_{\theta_j^\ell}[\log p(\mathcal{D} \mid \theta_j^\ell) + \log p(\theta_j^\ell)] + \nabla_{\theta_j^\ell} k(\theta_j^\ell, \theta)\right]$
    end for
  end for

Notice that:

1. The iteration over particles i is embaressingly parallel
2. Each of the n^2 updates requires computing a score $\log p(\mathcal{D} \mid \theta)$ over the full dataset $\mathcal{D}$

3 Distributed SVGD

The second observation motivates our first proposed method, which extends SVGD to the setting where $\mathcal{D} = \sqcup_{s=1}^S \mathcal{D}_s$ is partitioned across s different shards.

We make the following key assumption:

1. Data is iid, so we can decompose $\log p(\mathcal{D} \mid \theta) = \sum_{s=1}^S p(\mathcal{D}_s \mid \theta)$
2. The set of particles $\{\theta_i\}_{i=1}^n$ can fit on a single machine

Assumption 1 enables us to rewrite the computation for $\hat{\phi}^*(\theta)$ as

$$\hat{\phi}^*(\theta) = \frac{1}{n}\sum_{j=1}^n \left[k(\theta_j^\ell, \theta)\left[\sum_{s=1}^S \nabla_{\theta_j^\ell} \log p(\mathcal{D}_s \mid \theta_j^\ell) + \nabla_{\theta_j^\ell} \log p(\theta_j^\ell)\right] + \nabla_{\theta_j^\ell} k(\theta_j^\ell, \theta)\right] \tag{7}$$

Assumption 2 permits broadcasting of the set of particles θ_j across to each of the s workers, leading to a map-reduce style distributed algorithm

1. Master broadcast $\{\theta_j\}_{j=1}^n$ to all of the s workers
2. Each worker computes a local score $\nabla_{\theta_j^\ell} \log p(\mathcal{D}_s \mid \theta_j^\ell)$ (same dimensionality as $\{\theta_j\}_{j=1}^n$)
3. Results are grouped by particle θ_j and reduced (by summation) onto the master node, which performs the particle update

3.1 Analysis of communication complexity

We assume that communication occurs point-to-point, nodes have their own upstream and downstream links, and that a barrier synchronization occurs at the end of each iteration. Such assumptions are realistic in distributed computing settings such as cloud environments where issues with large datasets are typically encountered.

The communication complexity between two synchronization barriers is determined by the slowest node, so we focus attention on the bottleneck nodes. If θ is p-dimensional, the communication complexity of each iteration is

$$pn \times S + pn \times S \tag{8}$$

where the first term corresponds to the sequential broadcast of all particles through the master's upstream link and the latter to the reduction of all local scores through the master's downstream link.

A k-wise tree-broadcast and reducecan be applied to lower communication costs to $2pnk \log_k S$, but incurs an additional constant message serialization/deserialization overhead.

3.2 Partitioning particles to minimize communication bottleneck

The previous algorithm exhibits a communication bottleneck on the master node, resulting in a higher communication cost per iteration. To avoid this bottleneck, we can exploit our first observation that the iteration over particles is trivially parallelizable. Rather than designating a master node, partition the particles equally across all nodes. In doing so, the broadcast is converted into an `all_gather` and the reduce into an `all_reduce`.

By doing so, we avoid hot-spotting on the master and get communication cost

$$p(n/S)(S-1) + p(n/S)(S-1) + pnk \log_k S \approx 2pn + pnk \log_k S \tag{9}$$

3.3 Stochastic approximations

Treat partitions as subsampled minibatches (in reality we don't resample the minibatch)

$$\nabla_\theta \log p(\theta \mid \mathcal{D}) = \nabla_\theta \log p(\theta) + \nabla_\theta \log p(\mathcal{D} \mid \theta) \tag{10}$$

$$\approx \nabla_\theta \log p(\theta) + \frac{|\mathcal{D}|}{|\mathcal{D}_s|} \nabla_\theta \log p(\mathcal{D}_s \mid \theta) \tag{11}$$

Subsample the particle interactions using only local particles

$$\frac{1}{n} \sum_{j=1}^{n} \left[k(x_j, x) \nabla_{x_j} \log p(x_j) + \nabla_{x_j} k(x_j, x) \right] \approx \frac{1}{\Omega} \sum_{j \in \Omega} \left[k(x_j, x) \nabla_{x_j} \log p(x_j) + \nabla_{x_j} k(x_j, x) \right] \tag{12}$$

To prove: both LHS and RHS are empirical averages, concentration about mean.

If $\Omega \subset [n]$ is sampled iid with replacement, then both are iid samples of the data X. If $f \leq B$, then Hoeffding's inequality gives

$$P(\sum_{1}^{n} f(X_i) - \mathbb{E}f(X) \geq \alpha) \leq \exp\left(-\frac{\alpha^2}{2nB^2}\right) \tag{13}$$

4 Experiments

References

[1] Qiang Liu, Jason Lee, and Michael Jordan. "A kernelized Stein discrepancy for goodness-of-fit tests". In: *International Conference on Machine Learning*. 2016, pp. 276–284.

[2] Qiang Liu and Dilin Wang. "Stein variational gradient descent: A general purpose bayesian inference algorithm". In: *Advances In Neural Information Processing Systems*. 2016, pp. 2378–2386.