STOCHASTIC VARIATIONAL INFERENCE SURVEY

FINAL PROGRESS REPORT

KUSHAL KUMAR
14346

PRATIK MISHRA
14493

1 Problem statement

The objective of this project is to understand the theoretical aspects of Stochastic Variational Inference, comprehend recent research papers, apply them on actual data sets of problems proposed by these papers and analyze its performance.

2 Motivation

One of the major challenges of modern day statistician is to approximate intractable or difficult to compute probability densities, which is the core of Bayesian inference. Various techniques like MCMC sampling, Independence sampling, Gibbs and variational inference are used to approximate posteriors using appropriate proposal distribution.

Modern data analysis requires computation with massive data. Statistical machine learning research has addressed problems faced by data analysis by developing the field of probabilistic modeling, a field that provides an elegant approach to developing new methods for analyzing data (Pearl, 1988; Jordan, 1999; Bishop, 2006; Koller and Friedman, 2009; Murphy,2012)[1].

In particular, probabilistic graphical models give us a visual language for expressing assumptions about data and its hidden structure. The corresponding posterior inference algorithms let us analyze data under those assumptions, inferring the hidden structure that best explains our observations.

But the problem we face is scale. Inference algorithms of the 1990s and 2000s used to be considered scalable, but they cannot easily handle the amount of data that is generated and dealt with in modern times. This is the problem that Stochastic Variational Inference deals with. It is an approach of computing with graphical models that is appropriate for massive data sets, data that might not fit in memory or even be stored locally.

3 Theory

We have a family of densities Θ over some free variables called ”variational parameters” which we set suitably to solve an optimization problem to choose the best member of the family which approximates the posterior distribution $p(z|x)$. This is largely the governing idea of variational inference. We intend to minimize the *KL* divergence between the posterior and the distribution $q(z)$ from Θ to get the best approximation. But finding the *KL* divergence expression itself is intractable as the posterior is not known. This problem is solved by maximizing the Evidence Lower Bound (ELBO) which is equivalent to minimizing the *KL* divergence and this sets up the required optimization problem for the variational inference.

$$q^*(z) = \arg\max_{q \in \Theta}(\mathbb{E}[log(p(z,x))] - \mathbb{E}[log(q(z))])$$

Further to simplify the optimization problem we use the mean field assumption is which we assume that the latent variables are mutually independent and are governed by distinct factor in the proposal density i.e. $q(z) = \prod_{i=1}^{n} q_i(z_i)$, where each of $q_i(z_i)$ are called the "variational factors". Given the mean-field variational density one can iteratively optimize the ELBO using the technique called **coordinate ascent mean-field variational inference** (CAVI). From simple computations one gets that for each latent variable z_i, assuming the other latent variables z_{-i} and $q_l(z_l)$ to be known for $l \neq i$, one gets the optimal $q_i^*(z_i) \propto exp(\mathbb{E}_{-i}[log(p(z_i|z_{-i}, x))])$.

4 Stochastic Variational Inference

Stochastic Variational Inference builds on Variational Inference, a method that transforms complex inference problems into high-dimensional optimization problems (Jordan et al., 1999; Wainwright and Jordan,2008). Traditionally, Variational inference uses coordinate ascent algorithm which iterates between re-analyzing every data point in the data set and re-estimating its hidden structure. However, this practice becomes inefficient over large Datasets. **Stochastic Variational Inference** handles this problem by using stochastic optimization (Robbins and Monro, 1951), a technique that follows noisy estimates of the gradient of the objective. Noisy estimates of a gradient are often cheaper to compute than the true gradient, and following such estimates can allow algorithms to escape shallow local optima of complex objective functions. In statistical estimation problems, including variational inference of the global parameters, the gradient can be written as a sum of terms (one for each data point) and we can compute a fast noisy approximation by sub-sampling the data. With certain conditions on the step-size schedule, these algorithms provably converge to an optimum (Robbins and Monro, 1951)[1].

As one can realize that in CAVI in each iteration the algorithm goes through every data point, hence when the data size grows this approach for finding the optimal density from Θ will not be efficient. SVI solves this problem of large data size as it ascends ELBO at each iteration in the direction of the gradient. To accomplish that SVI uses global variational parameter λ, which is the parameter on which global latent variable β depends through $p(\beta|\lambda)$, in the conditionally conjugate model example model based on exponential family. Let each of the z_i depend on the local variational parameter ϕ_i and the posterior $p(z_i, x_i|\beta) = h(z_i, x_i)exp(\beta^T t(z_i, x_i) - A(\beta))$ belongs to the exponential family, and the prior on the global variables also belong to the exponential family, $p(\beta) = h(\beta)exp(\alpha^T[\beta, -A(\beta)] - A(\beta))$. We can now compute the natural parameters for the global variable, say a'. Then the Euclidean gradient,

$$\Delta_\lambda ELBO = A''(\lambda)(\mathbb{E}_\phi[a'] - \lambda)$$

So, the natural gradient is simply

$$g(\lambda) = \mathbb{E}_\phi[a'] - \lambda$$

This is the basic difference between the coordinate ascent updates and the SVI updates that lie along the natural gradient which is cheaper to compute and moving in different directions amounts to equal change for symmetrized *KL* divergence. This does not happen for Euclidean gradient. Now we can use simple gradient ascent to obtain parameter update equation,

$$\lambda_t = \lambda_{t-1} + \epsilon_t g(\lambda_t)$$

where ϵ_t is the step size. Note that this update procedure has to compute the expectation term hence still needs to go thought the entire data set. To overcome this problem, we use stochastic optimization of the ELBO. We know that,

$$g(\lambda) = \alpha + \left[\sum_{i=1}^{n} \mathbb{E}_{\phi_i^*}[t(z_i, x_i)], n\right]^T - \lambda$$

Instead of using this $g(\lambda)$ we would rather randomly choose a value of i say i_0 from $Unif(1, 2, ...n)$, and hence,

$$g(\lambda) = \alpha + \left[\mathbb{E}_{\phi_{i_0}^*}[t(z_{i_0}, x_{i_0})], 1\right]^T - \lambda$$

Though incorporating such a $g(\lambda)$ would make step directions arbitrary but it is much cheaper to compute and it ensures unbiasedness of natural gradient and hence convergence is guaranteed[2].

5 Probabilistic Models

5.1 Latent Dirichlet Allocation

5.1.1 Theory

LDA is a mixture model. It assumes that each document contains various topics, and words in the document are generated from those topics. All documents contain a particular set of topics, but the proportion of each topic in each document is different.

The generative process of the LDA model can be described as follows:

Given: Dirichlet distribution with parameter vector α of length K
Given: Dirichlet distribution with parameter vector β of length V

For topic number 1 to topic number K
draw a word distribution, i.e. a multinomial with parameter vector ϕ_k
according to β, ϕ taken from Dirichlet(β)

For document number 1 to document number M
draw a topic distribution, i.e. a multinomial with parameter vector θ
according to α, θ taken from Dirichlet(α)

For each word in the document
draw a topic z according to θ, z taken from Multinomial(θ)
draw a word w according to ϕ_z, w taken from Multinomial(ϕ_z)

In LDA, each document exhibits the same shared topics but with different proportions. LDA models an observed collection of documents,where each document is a collection of words words. Analyzing the documents amounts to posterior inference of p(β,θ,z|w). Conditioned on the documents, the posterior distribution captures the topics that describe them, the degree to which each document exhibits those topics, and which topics each word was assigned to. We can use the posterior to explore large collections of documents.

The posterior is intractable to compute (Blei et al., 2003). Approximating the posterior in LDA is a central computational problem for topic modeling. Researchers have developed many methods, including Markov chain Monte Carlo methods (Griffiths and Steyvers, 2004), expectation propagation (Minka and Lafferty, 2002), and variational inference (Blei et al., 2003; Teh et al., 2006b; Asuncion et al., 2009).

5.1.2 Problem and Data

The problem we deal with in our application is a classic example problem of LDA of finding the topic distribution in documents. For the dataset, we consider randomly downloaded Wikipedia articles, which is easy to generate thanks to python libraries and Wikipedia APIs,and apply the algorithm, which uses stochastic optimization to maximize the variational objective function for the LDA topic model. It only looks at a subset of the total corpus of documents each iteration, and thereby is able to find a locally optimal setting of the variational posterior over the topics more quickly than a batch VB algorithm could for large corpora.

5.1.3 Algorithm

```
Initialize $\lambda^{(0)}$ randomly.
Set the step-size schedule $\rho_t$ appropriately.
repeat
  Sample a document $w_d$ uniformly from the data set.
  Initialize $\gamma_{dk}$ = 1, for k ∈ {1,...,K}.
  repeat
    For n ∈ {1,...,N} set

          $\phi_{dn}^k \propto exp(\mathbb{E}[log(\theta_{dk})] + \mathbb{E}[log(\beta_{k,w_{dn}})])$, k ∈ {1,...,K}
```

```
      Set $\gamma_d = \alpha + \sum_n \phi_{dn}$
    until local parameters $\phi_{dn}$ and $\gamma_d$ converge.
    For k $\in$ {1,...,K} set intermediate topics

        $\hat{\lambda}_k = \eta + D \sum_{n=1}^{N} \phi_{dn}^{k} w_{dn}$

    Set $\lambda^{(t)} = (1 - \rho_t)\lambda^{(t-1)} + \rho_t \hat{\lambda}$.
until forever
```

5.1.4 Results

The output generated upon applying Stochastic Variational Inference on the LDA model to infer topic distribution in the wikipedia articles produced 20 topics (user defined parameter) that displayed the top 10 words along with their probabilites of occurence in each topic. Top 5 words of the first and second topics were as follows:

Topic 1:
based → 0.008211948289299923
information → 0.006985646976158434
language → 0.00665438506075405
computer → 0.006466267427718757
system → 0.006353878456587508

Topic 2:
convert → 0.07406781485578885
poland → 0.04939879690324016
gmina → 0.04457123764755828
athlete → 0.03828221742733228
flagiocathlete → 0.03560845403256021

5.2 Heirarchical Poisson Matrix Factorisation

Hierarchical Poisson Matrix Factorisation is a novel method to develop recommendation models based on implicit online feedback by the user such as views or clicks. It estimates users behaviour better than any competing methods as it captures the **long-tailed** user activity accurately. HPF learns the latent factors only through the positive examples which thereby removes the generation of negative examples from the picture. We develop the stochastically driven variational inference setup to approximate the posterior for HPF that scales upto large data.

We have the data about users and items, where each user has possibly rated a set of items they have used. The observation y_{ui} is the rating that user u gave to item i, or zero if no rating was given. Most of the values of the matrix y are zero.Here each item i is represented by a vector of K latent attributes β_i and each user u by a vector of K latent preferences θ_u. The observations y_{ui} are modeled with a Poisson distribution, parameterized by the inner product of the user preferences and item attributes, $y_{ui} \sim \text{Poisson}(\theta_u^T \beta_i)$.[3] More formally the HPF model is described as follows:
For each user u:

$$\text{Sample activity } \zeta_u \sim \text{Gamma}(a', a'/b')$$
$$\text{Sample preference } \theta_{uk} \sim \text{Gamma}(a, \zeta_u)$$

For each item i:

$$\text{Sample popularity } \eta_i \sim \text{Gamma}(c', c'/d')$$
$$\text{Sample attribute } \beta_{ik} \sim \text{Gamma}(c, \eta_i)$$

And for every user and item,

$$\text{Sample rating } y_{ui} \sim \text{Poisson}(\theta_u^T \beta_i)$$

Our goal is that given the user-item interaction matrix we need to infer the conditional distribution, $p(\theta, \beta|y)$. To approximate this we use SVI, given the knowlegde of all the prior and posterior distributions. For this we use the mean-field assumtion on the proposed distribution q over the latent factors as follows:

$$q(\beta, \theta, \zeta, \eta, z) = \prod_{i,k} q(\beta_{ik}|\lambda_{ik}) \prod_{u,k} q(\theta_{uk}|\gamma_{uk}) \prod_{u} q(\zeta_u|\kappa_u) \prod_{i} q(\eta_i|\tau_i) \prod_{i,u} q(z_{ik}|\phi_{iu})$$

Once we have approximated this distribution, we can use it for recommendations by estimating the posterior expectations of each user's expectations, each item attributes and thus form the recommendation for those items which the user has not used. We can also rank the each user's unused items by their posterior expected Poisson parameters[3].

The advantage of using HPF model over classical Gaussian Matrix Factorization are many:
1. HPF captures sparse factors due to the Gamma priors which encourages the most of the weight components to be close to zero by setting the shape parameter appropriately.
2. As mentioned before HPF captures long-tail of users and items, i.e. most of the users interact with a few items, hence we have large data for a few items, rest items are sparsely used by users. This fact is well illustrated in the paper [3] [Page - 3], where they have modeled HPF on Netflix data and then using the approximated posterior have generated some sample data, and have graphically illustrated how well the generated data resembles the long-tail behaviour as in the real data.

6 Edward

Edward is a Python library for probabilistic modeling, inference, and criticism. It is quick testing ground for fast experimentation and research with probabilistic models, ranging from classical hierarchical models on small data sets to complex deep probabilistic models on large data sets. Edward combines three fields: Bayesian statistics and machine learning, deep learning, and probabilistic programming.

It supports modeling with Directed graphical models,Neural networks (via libraries such as Keras and TensorFlow Slim),Intractable likelihoods,Bayesian nonparametrics and probabilistic programs.

It supports inference with Black box variational inference,Stochastic variational inference, Generative adversarial networks,Maximum a posteriori estimation, Hamiltonian Monte Carlo, Stochastic gradient Langevin dynamics,Gibbs sampling, Compositions of inference, Expectation-Maximization, Pseudo-marginal and ABC methods.

We implemented a Mixture model on Edward to infer hidden structure from unlabeled data, comprised of training examples. A mixture model is a model typically used for clustering. It assigns a mixture component to each data point, and this mixture component determines the distribution from which the data point is generated from. We implemented Gaussian Mixture model with the following likelihood:

$$p(x_n|\pi, \mu, \sigma) = \sum_{k=1}^{K} \pi_k Normal(x_n|\mu_k, \sigma_k)$$

and the following prior:

$$p(\pi) = Dirichlet(\pi|\alpha I_k) \quad p(\sigma_k^2) = InverseGamma(\sigma_k^2|a, b) \quad p(\mu_k) = Normal(\mu_k|0, I)$$

We generated a hand-made dataset by simulating random points from a multivariate Gaussian distribution assuming some values of the prior parameters. Using this data, we implemented our Gaussian Mixture Model to extract mixtures in the data. We also implemented topic extraction from wikipedia articles using Python packages, as the Edward impelementation was too slow for large dataset, which is already discussed under section **[5]** Probabilistic Models LDA.

7 More on SVI

7.1 Balanced Population SVI

In the real world, data is commonly imbalanced. For example, there are more images of cats and babies than other categories in social networks and there are more normal money transactions than fraudulent ones. Latent variable models that are used for tasks such as video summarization or fraud detection suffer from such imbalanced data. Aiming to maximize probability on the training data, they will infer redundant latent structures to capture the data clusters with high density but are not able to model the scarce data well that may contain possibly important information. For example, a latent space model may capture very fine differences between cats but might not even be able to model any feature of an animal with less pictures in the dataset. To extract more efficient latent representations of the underlying structure, there needs to be an approach based on biased subsampling of the original data set.

With large amounts of data and complex probabilistic models, stochastic variational inference with mini-batch subsampling is among the best available solutions for a broad range of applications.Sampling representative and balanced mini-batches is a desirable improvement and may result in more interpretable and less redundant latent structures, such as the topics in LDA.

To tackle the inference problem with imbalanced data, the mini-batches are sampled with Determinantal Point Process (DPP).The DPP is a probabilistic model that models random subsets of a ground set with repulsive interactions between the elements.The DPP is used to create diversified mini-batches of the data that are then used for the SVI updates.

It is assumed there is a balanced unknown population of datasets, and the observed data is an imbalanced realization from that population.Let θ denote latent variables of a Bayesian model. Suppose we sequentially observe S data points from the underlying population distribution as specified by the DPP, $X_S \sim DPP(X)$. This is the mini-batch. Every X_S induces a posterior $p(\theta|X_S)$, which is a function of the random data. Thus the required posterior is the population posterior:

$$p(\theta|DPP(\mathbf{X})) = \mathbb{E}_{\mathbf{X}_s \sim DPP(\mathbf{X})}[p(z, \beta|\mathbf{X}_s)]$$

[5] performed experiments to demonstrate BP-SVI on synthetic data with LDA. They generated a synthetic dataset following the generative process of LDA with a fixed global latent parameter (the graphical topics). They chose distinct patterns. To generate an imbalanced data set, they used different Dirichlet priors for the per document topic distribution θ. 300 documents were generated with prior (0.5 0.5 0.01 0.01 0.01); 50 with prior (0.01 0.5 0.5 0.5 0.01) and 10 with prior (0.01 0.01 0.01 0.5 0.5). Hence, the first two topics are used very often in the corpus. Topic 3 and 4 are shown a few times and topic 5 appears very rarely.

They applied LDA to recover the topics of the synthetic data using traditional SVI and their proposed BP-SVI respectively. Their aim was to recover the ground truth global parameter which indicates that the model is able to capture the underlying structure of the data. The results indicated that the first three topics were recovered using traditional SVI. Topic four was roughly recovered but with information from topic two mixed in. The last topic were not recovered at all. This shows the drawback of the tradition method that when the data is not balanced, the model create redundant topics to refine the likelihood of the dense data but ignore the scarce data. However, upon application of BP-SVI,all the topics were correctly recovered thanks to the balanced population.

7.2 Smoothed Gradients

Noisy stochastic gradients can slow down the convergence of SVI or lead to convergence to bad local optima. Hence, we propose a smoothing scheme to reduce the variance of the noisy natural gradient. To this end, we can smoothen the stochastic gradients to ensure faster convergence by averaging the sufficient statistics over the past iterations say L. Smoothed gradients are demonstrated over LDA model. Assume that $S(\lambda_i)$ be the sufficient statisti for the model, where λ_i are the global parameters on which the ELBO depends. Here is the sketch of the algorithm for smoothed gradient updates:

1. Uniformly sample a minibatch B_i \in {1, . . . , D} of documents. Compute the local variational parameters ϕ from a given λ_i.
2. Compute the sufficient statistics $\hat{S}_i = \hat{S}(\phi(\lambda_i), B_i)$.
3. Store $\hat{S}_i$, along with the L most recent sufficient statistics. Compute $\hat{S}_i^L = \frac{1}{L}\sum_{l=1}^{L-1} \hat{S}_{i-l}$ as their mean.
4. Compute the smoothed stochastic gradient according to
$$\hat{g}_i^L = (\eta - \lambda_i) + \hat{S}_i^L$$
5. Use the smoothed stochastic gradient to calculate $\lambda_i + 1$. Repeat.

The computatation of the smoothed gradient comes at no additional cost [6]. Here if we put $L = 1$, we get Stochastic Variational Inference with smooth gradient updates. Another advantage of using smooth gradient updates is that the variance of the gradients is approximately L times smaller that the original stochastic gradient [6]. Using such updates reduces the mean squared error relative to the full gradient. Proving that the algorithm is guaranteed to converge is still an open problem as the variational objective is non-convex makes it hard to prove.

7.3 Adaptive Learning Rate

Another aspect of SVI is to come up with an appropriate choice for the learning rate ϵ_t as described in the equations of the stochastic gradient updates. Research has been done on coming up with a suitable choice for this rate, for which [7] suggests a notion of *adaptive learning rate*. The adaptive learning rate is pulled by two signals. It grows when the current setting is expected to be far away from the coordinate optimal setting, but it shrinks with our uncertainty about this distance. We learn the learning rate by minimizing the expected error between the stochastic update and batch update. Let the squared norm of the error be

$$J(\epsilon_t) = (\lambda_{t+1} - \lambda_t^*)^T \times (\lambda_{t+1} - \lambda_t^*)$$

where, λ_t^* is the global parameter estimate from Batch updates. We obtain adaptive learning rate ϵ_t^* by minimizing the $\mathbb{E}_n[J(\epsilon_t|\lambda_t)]$. This leads to the stochastic update which is close in expectation to the batch update. Minimizing $\mathbb{E}_n[J(\epsilon_t|\lambda_t)]$ with respect to ϵ_t gives the following expression:

$$\epsilon_t^* = \frac{(\lambda_t - \lambda_t^*)^T(\lambda_t - \lambda_t^*)}{(\lambda_t - \lambda_t^*)^T(\lambda_t - \lambda_t^*) + tr(\Sigma)}$$

where $\sigma = Cov_n[\hat{\lambda}_t]$. However, this learning rate depends on unknown quantitiesthe batch update λ_t^* and the variance σ of the intermediate parameters around it. So, we estimate the adaptive learning rate within the SVI algorithm. Let g_t be the sampled natural gradient at t^{th} iteration, then $\mathbb{E}[g_t] = \lambda_t^* - \lambda_t$. Then we can write the espression for the adaptive learning rate as follows:

$$\epsilon_t^* = \frac{\mathbb{E}_n[g(t)]^T \mathbb{E}_n[g(t)]}{\mathbb{E}_n[g(t)^T g(t)]}$$

We can now form the Monte Carlo estimate of these updates at the time step t. Pluggin in the values we can estimate the adaptive learning rate stochastically. The computation requires no hand-tuning and uses computations already found inside stochastic inference. It works well for both stationary and non-stationary subsampled data. In their study of latent Dirichlet allocation, it led to faster convergence and a better optimum when compared to the best hand-tuned rates [7].

8 Road Ahead

Before we chart out the road ahead, we believe it is essential to summarize what we have learnt through this project. We got a first hand experience with handling software like edward that allows probabilistic modelling and inference, by implementing SVI to infer the appropriate posterior distribution. We also learnt models like Latent Dirichlet Allocation for extraction of topics from documents and Hierarchical Poisson Matrix Factorization for developing recommendation systems.Finally this project allowed us to visualize the power of Bayesian Models.

For our future work, we have observed that the lda implementation on Edward for small datasets was very easy and fast. However, Edward's lda implementation did not scale for large datasets and implementing lda on python and Tensorflow meant implementing algorithms on our own which could lead to bugs. The github repository for Edward is fairly active and people are actively trying to come up with a working implementation for lda on edward. We have looked at the issues and have decided to dedicate our efforts towards contributing in coming up with the said implementation.

References

[1] *Stochastic Variational Inference*: Matthew D. Hoffman, David M. Blei, Chong Wang and John Paisley.
`http://jmlr.org/papers/volume14/hoffman13a/hoffman13a.pdf`

[2] *Variational Inference: A Review for Statisticians*: David M. Blei, Alp Kucukelbir, Jon D. McAuliffe
`https://arxiv.org/pdf/1601.00670.pdf`

[3] *Scalable Recommendation with Hierarchical Poisson Factorization*: Prem Gopalan, Jake M. Hofman, David M. Blei
`http://jakehofman.com/inprint/poisson_recs.pdf`

[4] *Gaussian Processes for Big Data through Stochastic Variational Inference*: James Hensmen, Neil Lawrence
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.372.2036&rep=rep1&type=pdf`

[5] *Balanced Population Stochastic Variational Inference*: Cheng Zhang, Stephan Mandt
`http://approximateinference.org/accepted/ZhangEtAl2016.pdf`

[6] *Smoothed Gradients for Stochastic Variational Inference*: David M. Blei, Stephen Mandt
`https://papers.nips.cc/paper/5557-smoothed-gradients-for-stochastic-variational-inference.pdf`

[7] *An Adaptive Learning Rate for Stochastic Variational Inference*: David M. Blei, Rajnesh Ranganath, Chong Wang, Eric P. Xing
`http://proceedings.mlr.press/v28/ranganath13.pdf`