Iterative Refinement of Approximate Posterior for Training Directed Belief Networks

Hjelm R. D., Jojic, N., Cho, K., Chung, J., Salakhutdinov, R., and Calhoun, V.

November 10, 2015

Abstract

Deep directed graphical models, while representing a potentially powerful class of generative representations, are challenging to train due to difficult inference. Recent advances in variational inference, notably in variational autoencoders, have advanced well beyond traditional variational inference and MCMC methods. While these techniques offer higher flexibility as well as simpler and faster inference, they are still limited by the choice of the approximate posterior. Recently there has been interest in improving the approximate posterior through flexible priors and more involved inference procedures. We show that when the lower bound can be approximated as a deterministic function of the variational parameters, we can use standard gradient descent with back-propagation to improve the posterior and achieve a tighter lower bound. While our most notable results are with continuous latent variables, we show this can apply to binary latent variables as well with some success.

1 Introduction

Deep generative models offer the capacity for rich representations of complex data while (something about AI, reasoning, explaining away etc Neb?). Deep directed graphical models in particular have some advantage over their undirected cousins as they do not suffer from learning and evaluation issues arising from an intractable partition function. Despite their representational power, directed graphical models can be difficult to train, as the true posterior is typically intractable when the conditional has a complex or flexible parameterization, such as those modeled by a deep feed forward network with many layers of nonlinearities.

Recently, variational inference methods such as wake-sleep [5, 2], variational autoencoders (VAE) [8], and neural variational inference and learning (NVIL) [9] have supplanted traditional MCMC and traditional variational methods such as mean field coordinate ascent inference due to flexible and fast inference. This is accomplished by introducing a complex and flexible feed forward network as a parametric function for the approximate posterior. As opposed to traditional methods, these provide a fast and effective means of performing inference and learning which scale very well to large datasets.

While advancements have been significant, these methods are still limited by the choice of approximate posterior: the approximate nature of the posterior translates to a looseness in the lower bound. Some recent work as begun to address limitations in the posterior by either relaxing the factorial condition [3] on the approximate posterior or through the introduction of auxiliary networks. Alternatively, there is work focusing on improving the inference procedure itself, such as by applying gradient descent to variational inference [6]. Other work on MCMC variational inference improves the approximate posterior through iterative MCMC transitions [11]. We combine approaches of variational inference and gradient descent by augmenting variational inference starting at a parametric function defined by a deep feedforward network, using a gradient descent algorithm to improve inference. This way, we can sacrifice some computational efficiency in favor of a MCMC-like iterative procedure in order to improve the variational lower bound.

2 Gradient Backprop Variational Inference

Wake-sleep, VAE, and NVIL all optimize model parameters by minimizing the variational lower bound of the log likelihood:

$$\begin{aligned}\log p(x) &= \log \sum_h p(x,h) \\ &= \log \sum_h \tilde{q}(h|x) \frac{p(x,h)}{\tilde{q}(h|x)} \\ &\geq \sum_h \tilde{q}(h|x) \log \frac{p(x,h)}{\tilde{q}(h|x)} \\ &= \sum_h \tilde{q}(h|x) \log p(x,h) + \mathscr{H}(\tilde{q}) \\ &\approx \frac{1}{N} \sum_n \log p(x,h^{(n)}) - \log \tilde{q}(h^{(n)}), \end{aligned} \tag{1}$$

by introducing an approximate posterior $\tilde{q}_z$ with variational parameters z, where $\mathscr{H}(\tilde{q})$ is the entropy of the approximate posterior. Wake-sleep optimizes the approximate expectation of the joint density over samples from the approximate posterior / recognition net / inference net, while maximizing the approximate posterior over samples from the joint density. VAE maximizes both recognition and generation parameters through the lower bound via a clever re-parameterization, also using an MCMC approximation for the expectation of the conditional. In addition to taking samples from the approximate posterior, NVIL uses learned baselines to reduce variance when estimating model parameters.

We restrict ourselves to the case where there is one latent layer and prior is factorial. In order to augment our inference from those available from the parameterized approximate posterior, we wish to achieve a better posterior by taking a set number of iterative steps in the variational parameters, z. That is, given the approximate posterior has the form:

$$\tilde{q}(h|x;z) = \prod_i \tilde{q}(h_i|x;z), \tag{2}$$

we define the E-step loss function as

$$\mathscr{L} = \frac{1}{N} \sum_n \log p(x,h^{(n)}) - \log \tilde{q}(h^{(n)}), \tag{3}$$

such that we infer the variational parameters by applying iterative steps of gradient descent:

$$\begin{aligned}\nabla_z \mathscr{L} &= \frac{1}{N} \nabla_z \big(\sum_n \log p(x,h^{(n)}) - \log \tilde{q}(h^{(n)}) \big) \\ z_t &= z_{t-1} + \gamma \nabla_z \mathscr{L} \end{aligned} \tag{4}$$

where γ is the "inference rate".

Readers will probably notice immediately that the gradient in not tractable due to the stochastic variables used to calculate the expectation. However, we will show below that some straightforward to clever approximations can yield a *deterministic* function w.r.t. the variational parameters, thus making the gradient descent quite easy.

It may be that the number of inference steps required to infer a good approximate posterior may be prohibitively high, and we have some flexibility in the initial parameters z_0. So as to speed up learning, and because there is a chance that this optimization may be non-convex, we initialize the variational parameters given by a parametric function $z_0 = q_\phi(h|x)$ defined by a feed forward network. If we optimize the auxiliary network to minimize the KL divergence between the output of the network and the approximate posterior found in our gradient descent procedure, we reach variational inference in the limit of no additional gradient steps.

3 Continuous Latent Variables

Let us assume that our approximate posterior and prior distributions are parameterized by multivariate normal distributions with diagonal covariance. As in VAE, the KL divergence $KL(\tilde{q}(h|x)||p(h))$ can be calculated exactly, but the conditional term $\langle p(x|h) \rangle_{\tilde{q}}$ must be approximated by sampling from $\tilde{q}$.

In order to pass the gradient of the approximate log conditional to the variational parameters, we can re-parameterize our approximate posterior by:

$$h = \mu + \varepsilon \odot \sigma \tag{5}$$

where $(\mu, \sigma) = z$ are the mean and variance parameters for the posterior density and $\varepsilon \sim \mathcal{N}(0,1)$. The gradients of the approximate lower bound w.r.t. the variational parameters z can be calculated easily using back propagation.

4 Training Procedure

Much like traditional variational inference, our training procedure follows an EM algorithm.

4.1 E Step: Approximately Inferring $p(h|x)$

Our inference procedure is equivalent to

$$\begin{aligned}&\arg\max_q \sum_h \tilde{q}(h|x)\log p(x,h) + \mathscr{H}(\tilde{q}) \\ =&\arg\max_z \sum_h \tilde{q}(h|x;z)\log p(x,h) + \mathscr{H}(\tilde{q}).\end{aligned}$$

4.2 M Step: Estimating the Parameters θ and ϕ

Since the approximate inference has been computed, it is rather straightforward to compute the conditional term of the gradient of Eq. (1):

$$\nabla_\theta \langle \log p_\theta(x|h)\rangle_{\tilde{q}} \approx \frac{1}{N}\sum_n \nabla_\theta \log p_\theta(x|h^{(n)})$$

where $h^{(n)} \sim \tilde{q}(h|x)$ are samples from the approximate posterior with the parameters $z^\star$ obtained from the E-step. In order to make our inference procedure more efficient, we also optimize the auxiliary parameters ϕ of the auxiliary network $q_\phi(h|x)$ by minimizing:

$$C_q(\phi) = KL(\tilde{q}(h|x)||q_\phi(h|x)). \tag{6}$$

5 Binary Latent Variables

Unfortunately, the parameterization available to us with continuous latent variables with multivariate normal distributions is not available to us with binary latent variables. In order to use gradient descent to back-propagate the gradients of the lower bound, we need to make a stronger approximation than the one used above.

There has been some recent work into passing gradients through binary latent variables. Notably, we try two methods: one, we pass the centers of the Bernoulli distribution rather than sampling. That is, our loss becomes:

$$\mathscr{L}_e = -(\log p(x|\mu) + \langle \log p(h)\rangle_{\tilde{q}} - \langle \log \tilde{q}\rangle), \tag{7}$$

where $\log p(x|\mu)$ is the log-output of the generation network using the variational parameters $\mu = sigmoid(z)$. Note that the prior and entropy terms can be computed exactly without sampling.

Another option we explore here is to use the *straight through* (ST) [1] estimator introduced in [4]. This amounts to using the MCMC approximation for the log conditional and approximating the gradient of w.r.t the variational parameters as:

$$\begin{aligned}\frac{\partial \mathscr{L}_e}{\partial z} = &-\frac{1}{N}\sum_n \frac{\partial}{\partial h^{(n)}}\log p(x|h^{(n)}) \\ &-\frac{\partial}{\partial z}\big(\langle \log p(h)\rangle_{\tilde{q}} + \mathscr{H}(\tilde{q})\big)\end{aligned} \tag{8}$$

Finally, rather than taking parametric steps and using gradient steps, we use adaptive importance sampling to adjust the center of the Bernoulli distribution at each inference step:

$$\begin{aligned}h^{(n)} &\sim \mu_t \\ w^{(n)} &= p(x,h^{(n)})/q(h^{(n)}|x) \\ \tilde{w}^{(n)} &= w^{(n)}/\sum_n w^{(n)} \\ \mu_{t+1} &= \sum_n \tilde{w}^{(n)} * h^{(n)}\end{aligned} \tag{9}$$

6 Related Work

Our work combines many elements found in variational inference and MCMC methods. In spirit, it is closest to MCMC variational inference, yet it does not make use of an auxiliary network to perform the MCMC transitions, thus does not require extra parameters nor parameterization choice in inference outside of the normal inference

network. In parameterization then it is very similar to VAE, wake-sleep, and NVIL, assuming a factorial prior.

Recent work on stochastic variational inference [6] also uses gradient descent to improve on the variational inference algorithm. However, their work is limited to the exponential family and do not make use of the MCMC approximation for the log conditional, nor a feed forward network to parameterize inference.

7 Experimental Results

We test our model on MNIST, using the binarized dataset found in [10], with a standard train, validation, and test split of 50k, 10k, and 10k samples. Unfortunately, recent papers on VAE [9, 11] have used different, non-standard splits, or actively sample from continuous MNIST [3], making it difficult to compete. We however make sure that our results are from the most conservative setting (having the least amount of training examples), thereby making any comparison as meaningful as possible.

All models were trained with early stopping by recording best validation, and none of our models used any of the regularizations available to us, such as dropout, weight noise, or L2-norm regularization. All of our models were trained for at most 500 epochs, and each took no more than 24 hours to complete, best validation was typically reached much faster. All of our models suffered from overfitting at some point.

7.1 Continuous Variables

For the continuous latent variable case we used the same network structure as in [8, 11], with 200 continuous latent variables and a generation and recognition network with 500 deterministic softplus units. Each of our models were trained using 20 samples during inference and 20 samples to approximate the lower bound during the M-step. Parameters were adjusted using the rmsprop algorithm [4] with an initial learning rate of 3×10^{-3} with a batch size of 100 as per validation.

Our results with continuous variables are presented in table 7.1. For comparison are VAE results from [8] and Hamiltonian MCMC variational inference [11] with the number of leapfrog steps specified.

Model	lower bound	NLL
VAE	99	
$GBVI_{20}$	92.83	90.51
HVI_5[1]	90.86	87.16
HVI_{10}[1]	87.60	85.56

Table 1: Lower bounds and NLL for various continuous latent variable.

Model	lower bound	NLL
Wake-sleep	120.7	116.3
NVIL	113.1	
AVI_{20}	114.08	110.13
$AVI_{20}(240)$	107.03	103.10
$MPVI_{20}$		
$MPVI_{20}(240)$	122.67	103.32

Table 2: Test lower bounds and NLL for various continuous latent variable models.

7.2 Binary Variables

For binary latent variables, we used two structures to compare to wake-sleep and NVIL. For comparison to wake-sleep, we used a shallow feed forward network with 200 output neurons for the binary latent variables for the posterior and no deterministic units for the posterior nor conditional. As NVIL uses an additional feed forward network for baseline prediction, we used an deterministic layer of 240 softplus units to roughly match the total number of parameters. Parameters were adjusted using Adam [7] with an initial learning rate of 3×10^{-4} with a batch size of 10 as per validation.

8 Discussion

todo

References

[1] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neu-

rons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[2] J. Bornschein and Y. Bengio. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751*, 2014.

[3] Y. Burda, R. Grosse, and R. Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

[4] G. Hinton. Neural networks for machine learning. Coursera, video lectures, 2012.

[5] G. Hinton, P. Dayan, B. Frey, and R. Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

[6] M. Hoffman, D. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[7] D. Kingma and J. B. Adam. A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[8] D. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[9] A. Mnih and K. Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

[10] R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879. ACM, 2008.

[11] T. Salimans. Markov chain monte carlo and variational inference: Bridging the gap. *arXiv preprint arXiv:1410.6460*, 2014.