JOINT STOCHASTIC APPROXIMATION LEARNING OF HELMHOLTZ MACHINES

Haotian Xu , Zhijian Ou
Department of Electronic Engineering
Tsinghua University
Beijing, 100084, China
xht13@mails.tsinghua.edu.cn,ozj@tsinghua.edu.cn

ABSTRACT

Though with progress, model learning and performing posterior inference still remains a common challenge for using deep generative models, especially for handling discrete hidden variables. This paper is mainly concerned with algorithms for learning Helmholz machines, which is characterized by pairing the generative model with an auxiliary inference model. A common drawback of previous learning algorithms is that they indirectly optimize some bounds of the targeted marginal log-likelihood. In contrast, we successfully develop a new class of algorithms, based on stochastic approximation (SA) theory of the Robbins-Monro type, to directly optimize the marginal log-likelihood and simultaneously minimize the inclusive KL-divergence. The resulting learning algorithm is thus called joint SA (JSA). Moreover, we construct an effective MCMC operator for JSA. Our results on the MNIST datasets demonstrate that the JSA's performance is consistently superior to that of competing algorithms like RWS, for learning a range of difficult models.

1 INTRODUCTION

Deep generative models (DGMs) have the potential to learn representation and capture high-level abstractions for high-dimensional data(Hinton et al., 2006; Bengio, 2009). DGMs, often involving multiple layers of hidden variables $\boldsymbol{h}$ in addition to observed variables $\boldsymbol{x}$, are highly expressive. However, model learning and performing posterior inference still remains a common challenge for using those DGMs, especially for handling discrete hidden variables[1].

Recently, there are increasing interests in pairing the generative model with an auxiliary inference model which approximates the posterior inference for the generative model. During training, it is not only to fit the the generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})$ parameterized by $\boldsymbol{\theta}$, to the training data, but also to simultaneously train the inference model $q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})$ parameterized by $\boldsymbol{\phi}$. This basic idea has been proposed and enhanced many times - initially the Helmholtz machine with the wake-sleep algorithm (WS)(Hinton et al., 1995; Dayan et al., 1995); and more recently the variational autoencoder (VAE)(Kingma et al., 2013), the deep latent Gaussian models (DLGM) with stochastic backpropagation(Rezende et al., 2014), neural variational inference and learning (NVIL)(Mnih & Gregor, 2014), re-weighted wake-sleep (RWS)(Bornschein & Bengio, 2014), importance weighted auto-encoders (IWAE)(Burda et al., 2015), and so on. While the above basic idea remains essentially unchanged, there are much progress in inference and learning algorithms for Helmholtz machines.

Generally speaking, for those models with a pair of generative and inference models, we could call them Helmholtz machines. This paper is mainly concerned with algorithms for learning such Helmholz machines. We leave the detailed discussion of various existing algorithms to Related Work Section, where we review three main classes of learning algorithms, depending on the objective functions used in joint training of p and q model. Remarkably, a common drawback of all the

[1] since the reparameterization trick (Kingma et al., 2013) already provides an effective method for handling continuous hidden variables as we discuss below.

three class of algorithms is that they indirectly optimize some bounds of the targeted marginal log-likelihood, whether the variational approximated bound or the IS approximated bound. Accordingly, the role of the auxiliary inference model in training is the approximation distribution in variational approximation or the proposal distribution in IS.

In contrast to these previous approaches, we propose a new class of algorithms for learning Helmholtz machines, based on stochastic approximation (SA) theory of the Robbins-Monro type Robbins & Monro (1951). Basically, the Robbins-Monro algorithm is a methodology for solving a root finding problem, where the function is represented as an expected value. The SA algorithm iterates Markov move (implemented via Markov Chain Monte Carlo, MCMC) and parameter update. The convergence of SA has been studied under various conditions (Benveniste et al., 1990; Chen, 2002; Tan, 2015).

We first show how to directly optimize the marginal log-likelihood and simultaneously minimize the inclusive KL-divergence in the framework of SA. The key is to formulate two gradients as expectations and equate them to zero, then SA is ready to solve the two simultaneous equations. It can be proved that under mild conditions, the iterative updated parameters converges to the roots of the equations almost surely. The resulting learning algorithm is thus called joint SA (JSA). Second, in the SA setting, the role of the inference model becomes the proposal distribution for constructing MCMC operator. Specifically, we construct two types of MCMC operators - one is based on the standard metropolis independence sampler (MIS)(Hastings, 1970) and the other is based on the multiple-trial metropolis independence sampler (MTMIS)(Liu, 2008), which considerably improve SA convergence as shown in our experiments.

The JSA learning algorithm can handle both continuous and discrete variables. In this paper, we mainly present experimental results for training discrete belief networks with discrete Bernoulli and multinomial variables for unsupervised learning tasks. Our results on the MNIST datasets demonstrate that the JSA's performance is consistently superior to that of competing algorithms like RWS, for learning a range of difficult models.

2 JOINT STOCHASTIC APPROXIMATION (JSA)

2.1 STOCHASTIC APPROXIMATION BACKGROUND

Since Robbins & Monro (1951), a vast theoretical literature has grown up, concerning conditions for convergence, rates of convergence, multivariate, proper choice of step size, and so on (Benveniste et al., 1990; Chen, 2002). Recently, it has been shown to work surprisingly well when training complex generative models, including restricted Boltzmann machines (Tieleman, 2008), deep Boltzmann machines (Salakhutdinov & Hinton, 2009), and trans-dimensional random fields (Wang et al.).

Basically, the SA algorithm is a methodology for solving a root finding problem, where the functions are represented as expected values. Suppose that the objective is to find a solution $\boldsymbol{\varphi}^*$ to $h(\boldsymbol{\varphi}) = 0$ with

$$h(\boldsymbol{\varphi}) = E_{\boldsymbol{\varphi}}\left[H(\boldsymbol{Y}; \boldsymbol{\varphi})\right] \tag{1}$$

where $\boldsymbol{\varphi} \in R^d$ is the parameter of dimension d, $H(\boldsymbol{Y}; \boldsymbol{\varphi}) \in R^d$ is the d-dimensional noisy statistics and $E_{\boldsymbol{\varphi}}$ denotes the expectation with respect to $\boldsymbol{Y} \sim f(\cdot; \boldsymbol{\varphi})$, a probability density function depending on $\boldsymbol{\varphi}$. A general SA algorithm is as follows.

Stochastic Approximation (Tan, 2015):

- Generate $\boldsymbol{Y}^{(t)} \sim K_{\boldsymbol{\varphi}^{(t-1)}}(\boldsymbol{Y}^{(t-1)}, \cdot)$, a Markov transition kernel admits $f(\cdot; \boldsymbol{\varphi}^{(t-1)})$ as the invariant distribution.
- Set $\boldsymbol{\varphi}^{(t)} = \boldsymbol{\varphi}^{(t-1)} + \alpha_t H(\boldsymbol{Y}^{(t)}; \boldsymbol{\varphi}^{(t-1)})$.

The convergence of SA has been studied under mild conditions, e.g. the learning rates α_t satisfying that $\sum_{t=0}^{\infty} \alpha_t = \infty$ and $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$. In practice, we can use large learning rates at the early stage of learning and switch to $\frac{1}{t}$ for convergence.

2.2 DEVELOP JSA FOR LEARNING HELMHOLTZ MACHINES

The Helmholtz machine is characterized by pairing a generative model $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})$ with an auxiliary inference model $q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})$. Based on SA theory of the Robbins-Monro type (Robbins & Monro, 1951), the proposed JSA algorithm is to estimate the two set of parameters — $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, by directly optimizing the marginal log-likelihood and simultaneously minimizing the inclusive KL-divergence.

To that end, we need to solve the simultaneous equations:

$$\begin{cases} \dfrac{\partial}{\partial \boldsymbol{\theta}} log p_{\boldsymbol{\theta}}(\boldsymbol{x}) = & E_{p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})}\left[\dfrac{\partial}{\partial \boldsymbol{\theta}} log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})\right] = 0 \\ \dfrac{\partial}{\partial \boldsymbol{\phi}} KL(p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})||q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})) = & -E_{p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})}\left[\dfrac{\partial}{\partial \boldsymbol{\phi}} log q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})\right] = 0 \end{cases} \tag{2}$$

which exactly follows the form of Equ. 1. Applying the SA algorithm, we obtain the SA recursion for updating parameters:

$$\begin{cases} \boldsymbol{\theta}^{(t)} = & \boldsymbol{\theta}^{(t-1)} + \alpha_t \dfrac{\partial}{\partial \boldsymbol{\theta}} log p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}^{(t)}) \\ \boldsymbol{\phi}^{(t)} = & \boldsymbol{\phi}^{(t-1)} + \beta_t \dfrac{\partial}{\partial \boldsymbol{\phi}} log q_{\boldsymbol{\phi}}(\boldsymbol{h}^{(t)}|\boldsymbol{x}) \end{cases} \tag{3}$$

where $\boldsymbol{h}^{(t)}$ is drawn from the MCMC transition kernel that admits $p_{\boldsymbol{\theta}^{(t-1)}}(\boldsymbol{h}|\boldsymbol{x})$ as the invariant distribution.

For this purpose, we use the inference model as the proposal distribution to construct the MCMC operator. Note that as during the SA recursions, the inclusive KL-divergence between the target posteriori distribution $p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})$ and the proposal distribution $q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})$ is to be minimized. This means that the quality of the proposal distribution will be optimised. Moreover, the inclusive KL-divergence tends to find proposal distributions that have higher entropy than the original which is advantageous for sampling (the exclusive KL would be unsuitable for this reason, (MacKay, 2003)).

In the following, we construct two types of MCMC operators - one is based on the standard metropolis independence sampler (MIS)(Hastings, 1970) and the other is based on the multiple-trial metropolis independence sampler (MTMIS)(Liu, 2008), which considerably improve SA convergence as shown in our experiments.

2.2.1 METROPOLIS INDEPENDENCE SAMPLER (MIS)

With abuse of notation, suppose the target distribution is $\pi(\boldsymbol{x})$. MIS uses a proposal distribution $g(.)$, independent of the previous state.

The MIS Scheme: given the current state $\boldsymbol{x}^{(t)}$

- Draw $\boldsymbol{y} \sim g(\boldsymbol{y})$.
- Set $\boldsymbol{x}^{(t+1)} = \boldsymbol{y}$ with probability

$$min\left\{1, \frac{w(\boldsymbol{y})}{w(\boldsymbol{x}^{(t)})}\right\}, w(\boldsymbol{x}) = \frac{\pi(\boldsymbol{x})}{g(\boldsymbol{x})} \tag{4}$$

2.2.2 MULTIPLE-TRIAL METROPOLIS INDEPENDENCE SAMPLER (MTMIS)

While MIS is simple and easy to implement, it may mix slowly. MTMIS can make large-step moves along the favorable directions which can accelerate the mixing speed.

The MTMIS Scheme: given the current state $\boldsymbol{x}^{(t)}$

- Generate K i.i.d samples $\boldsymbol{y}_j \sim g(\boldsymbol{y}), j = 1, \ldots, K$, compute $w(\boldsymbol{y}_j) = \pi(\boldsymbol{y}_j)/g(\boldsymbol{y}_j)$ and $W = \sum_{j=1}^{k} w(\boldsymbol{y}_j)$
- Draw $\boldsymbol{y}$ from the trial set $\{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$ with probability proportional to $w(\boldsymbol{y}_j)$

- Set $\boldsymbol{x}^{(t+1)} = \boldsymbol{y}$ with probability

$$min\left\{1, \frac{W}{W - w(\boldsymbol{y}) + w(\boldsymbol{x}^{(t)})}\right\} \tag{5}$$

In the case of learning Heltomhltz machines, we use the inference model $q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})$ as the trial distribution to propose samples for $p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})$. The importance sampling weight is calculated as :

$$w(\boldsymbol{h}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})}{q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})} \propto \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})}{q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})} \tag{6}$$

Algorithm 1 JSA Learning of Helmholtz machines

1: **for** number of training iterations **do**
2: Randomly pick a training sample $\boldsymbol{x}$ along with its cached hidden states of each layer $\boldsymbol{h}_{\boldsymbol{x}}$; Generate sample $\boldsymbol{h}$ via MIS or MTMIS;
3: Update parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ based on Equ. 3;
4: Update cached hidden states $\boldsymbol{h}_{\boldsymbol{x}}$ for $\boldsymbol{x}$ using samples from $q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})$;
5: Adjust the learning rate.
6: **end for**

3 Related Work

Depending on the objective functions used in joint training of p and q model, there exist three main classes of algorithms for learning Helmholtz machines, as summerized in Table. 1.

The first class (e.g. VAE, stochastic backpropagation, NVIL, MuProp), uses the variational lower bound of the marginal log-likelihood as the single objective function, which contains the parameters of both models $(\boldsymbol{\theta}, \boldsymbol{\phi})$. While the gradient with respect to generative parameter $\boldsymbol{\theta}$ is well-behaved and does not pose a problem, the gradient with respect to inference parameter $\boldsymbol{\phi}$ is known to have high variance. To address this problem, reparameterization trick Kingma et al. (2013) has been developed, providing an effective method for handling continuous hidden variables. Exploring variance reduction techniques, e.g. NVIL and MuProp(Gu et al., 2015), still remains a major challenge for this class of algorithms, especially for handling discrete hidden variable, such as Bernoulli or multinomial.

Another class (e.g. WS, RWS), uses two objective functions - the importance sampling (IS) approximated lower bound of the marginal log-likelihood [2] for optimizing $\boldsymbol{\theta}$ and the inclusive KL-divergence $KL(p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})||q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})))$ for optimizing $\boldsymbol{\phi}$. The motivation is mixed. Learning through many cycles of sensing and dreaming seems to be biological attractive. In addition, noting that the optimization of the variational bound with respect to $\boldsymbol{\phi}$ amounts to minimize the exclusive KL-divergence $KL(q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{h}|\boldsymbol{x})))$, which has the undesirable effect of high variance mentioned before. Technically, update $\boldsymbol{\phi}$ by optimizing the inclusive KL-divergence would be better than optimizing the exclusive KL-divergence. A shortcoming of this class of algorithms is that both the IS estimated gradients for $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are biased estimators. Moreover, the heuristic use of the stochastic gradients derived from two objective functions also does not provide convergence guarantees.

The third class, mainly the IWAE algorithm, also use a single objective function - the IS approximated lower bound (inspired from RWS). So IWAE is much close to the first class of learning algorithms. By use of the reparameterization trick (inspired from VAE), IWAE can only be applied to DGMs consisting of continuous latent variables.

[2] The IS approximated bound is slightly different from the variational bound. The difference is that the former uses importance weighted averaging in Monte Carlo estimation. When using one sample in Monte Carlo estimation, they two are equivalent.

Table 1: Review of existing algorithms for learning Helmholtz machines. For each algorithm, we list the objection function optimized to learn the p and q model (ML: maximum likelihood, V-LB : the variational lower bound, IS-LB : the IS approximated lower bound), and the supported types of random variables (RV) (C: continuous, D: discrete).

Algorithm		$p_\theta(\boldsymbol{x}, \boldsymbol{h})$			$q_\phi(\boldsymbol{h}\|\boldsymbol{x})$		RV type	
		ML	V-LB	IS-LB	$KL(q\|\|p)$	$KL(p\|\|q)$	C	D
1	VAE (Kingma et al., 2013)		$\surd$		$\surd$		$\surd$	
	NVIL (Mnih & Gregor, 2014)		$\surd$		$\surd$		$\surd$	$\surd$
	MuProp (Gu et al., 2015)		$\surd$		$\surd$		$\surd$	$\surd$
2	WS (Hinton et al., 1995)		$\surd$			$\surd$	$\surd$	$\surd$
	RWS (Bornschein & Bengio, 2014)			$\surd$		$\surd$	$\surd$	$\surd$
3	IWAE (Burda et al., 2015)			$\surd$	$\surd$		$\surd$	
JSA		$\surd$				$\surd$	$\surd$	$\surd$

4 EXPERIMENTS

4.1 SETTINGS

We conduct the experiments on the MNIST dataset with a standard training, validation and testing split of 50k, 10k, and 10k samples. We use the MNIST dataset which was binarized according to Murray & Salakhutdinov (2009). Both the generative and inference models are of the same type - either sigmoid belief networks (SBNs) (Saul et al., 1996) with discrete Bernoulli variables or categorical SBNs with multinomial variables.

For training, we use stochastic gradient decent without momentum and set mini-batch size to 100. The experiments were run with learning rates of $\{0.0005, 0.001\}$. From these two we always report the result with the highest validation log-likelihood.

For estimating marginal log-likelihood, we use the same method in RWS (Bornschein & Bengio, 2014), namely

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{\boldsymbol{h}} q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x}) \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h})}{q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x})} \simeq \frac{1}{M} \sum_{m=1}^{M} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{h}^{(m)})}{q_{\boldsymbol{\phi}}(\boldsymbol{h}^{(m)}|\boldsymbol{x})}, \boldsymbol{h}^{(m)} \sim q_{\boldsymbol{\phi}}(\boldsymbol{h}|\boldsymbol{x}) \tag{7}$$

We use 100 samples to monitor log-likelihood on validation set and use 100,000 samples to estimate log-likelihood on test set based on Equ. 7.

For our own run of RWS (using the authors' code) for training categorical SBN, we use the same experimental settings with Bornschein & Bengio (2014) including the learning rate, momentum and the number of the samples used (i.e. 10) during training. For fair comparison of computational complexity, we also use 10 trials for each MTMIS markov move.

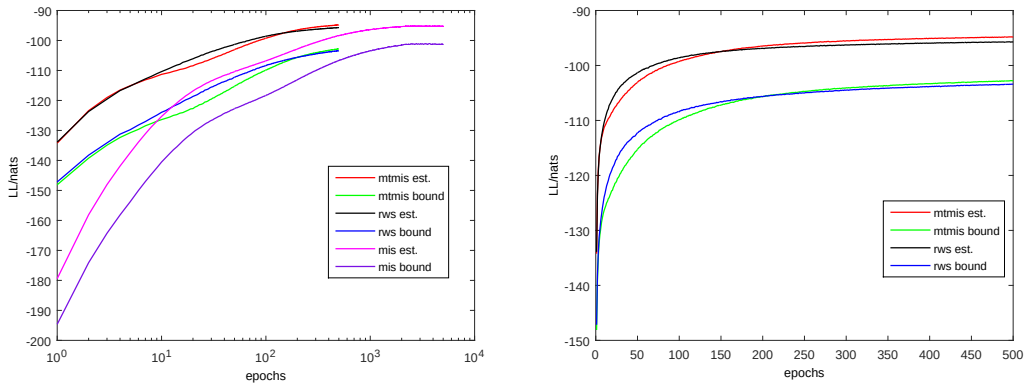
4.2 RESULTS

As can been seen from Table. 2, the JSA-MTMIS's performance (in terms of test likelihoods) is consistently superior to that of RWS, with the same computational cost, for learning a range of different models. Moreover, as shown in Fig. 1b, JSA-MTMIS has the same convergence speech as RWS.

From Fig. 1a, we can see that the JSA-MIS is much slower than JSA-MTMIS and RWS. When they three converge to a similar log-likelihood value, JSA-MIS takes 10 times epoches compared to the other two algorithms. It is due to the low sampling efficiency of MIS, since each Markov move in MIS just use one sample from the trial distribution $q(\boldsymbol{h}|\boldsymbol{x})$. In contrast, MTMIS draws 10 trial samples from $q(\boldsymbol{h}|\boldsymbol{x})$ and randomly chooses one, which could enable larger move and improve mixing. This can also be seen from the acceptance ratio during training. In JSA-MIS, 40-50% samples are accepted, while almost 80-90% samples are accepted in JSA-MTMIS.

Table 2: The negative log-likelihoods (NLL) on test dataset using 100,000 samples to estimate marginal log-likelihood based on Equ. 7 and lower bound. Both the generative and inference models are of the same type - either SBNs with discrete Bernoulli variables or categorical SBNs with multinomial variables (denoted by C). The results of * is from Bornschein & Bengio (2014), $\diamond$ are cited from Mnih & Gregor (2014), $\diamondsuit$ from Gu et al. (2015).

Model	200	200-200	200-200-200	200-10(C)	200-200-10(C)	200-200-200-10(C)
	NLL est. (NLL bound)					
WS	116.3^* (120.7^*)	106.9^* (109.4^*)	101.3^* (104.4^*)			
RWS	103.1^* —	93.4^* —	90.1^* —	97.65 (109.41)	90.35 (99.71)	88.43 (96.09)
JSA-MIS	103.5 (112.7)	93.33 (101.00)	89.85 (97.04)	97.8 (106.83)	91.60 (98.04)	88.43 (96.09)
JSA-MTMIS	102.3 (116.37)	92.11 (101.88)	88.92 (98.20)	97.05 (110.39)	89.84 (98.93)	87.82 (96.58)
NVIL	— ($113.1^\diamond$)	— ($99.8^\diamond$)	— ($96.7^\diamond$)			
MuProp	— ($113.1^\diamondsuit$)	— ($100.4^\diamondsuit$)	— ($98.6^\diamondsuit$)	— ($107.8^\diamondsuit$)		



(a) (b)

Figure 1: (a) The convergence curves of MTMIS, RWS and MIS for SBN-200-200 over the test dataset using 100 samples to estimate the marginal log-likelihoods and lower bounds. (b) The curves for the first 500 epochs taken from (a), which give a clear comparision between JSA-MTMIS and RWS

5 Conclusion

We introduce a new class of algorithms for learning Helmholtz machines - JSA. It directly maximizes the marginal likelihood and simultaneously minimizes the inclusive KL divergence. In the JSA setting, we treat the inference model as the trial distribution and construct two types of MCMC operators - MIS and MTMIS to sample from the true posterior. Our experiments demonstrate that JSA-MTMIS achieves better likelihood results on MINIST when compared to RWS. Besides, we also shows that MTMIS is more efficient and mixes more quickly than MIS. In the future, we would apply JSA to semi-supervised learning of Helmholtz machines.

ACKNOWLEDGMENTS

This project is supported by NSFC grant 61473168. We would like to thank Zhiqiang Tan for helpful discussions and the developers of Theano (Bergstra et al., 2010; Bastien et al., 2012) for their powerful software.

REFERENCES

Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive algorithms and stochastic approximations*. New York: Springer, 1990.

Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. *arXiv preprint arXiv:1406.2751*, 2014.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Hanfu Chen. *Stochastic approximation and its applications*. Springer Science & Business Media, 2002.

Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop unbiased backpropagation for stochastic neural networks. *arXiv preprint arXiv:1511.05176*, 2015.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The" wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Kingma, Diederik P, and Welling Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Jun S Liu. *Monte Carlo strategies in scientific computing*. Springer Science & Business Media, 2008.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

Iain Murray and Ruslan R Salakhutdinov. Evaluating probabilities under high-dimensional latent variable models. In *Advances in neural information processing systems*, pp. 1137–1144, 2009.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Ruslan Salakhutdinov and Geoffrey E Hinton. Deep boltzmann machines. In *International conference on artificial intelligence and statistics*, pp. 448–455, 2009.

Lawrence K Saul, Tommi Jaakkola, and Michael I Jordan. Mean field theory for sigmoid belief networks. *Journal of artificial intelligence research*, 4(1):61–76, 1996.

Zhiqiang Tan. Optimally adjusted mixture sampling and locally weighted histogram analysis. *Journal of Computational and Graphical Statistics*, 2015.

Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pp. 1064–1071. ACM, 2008.

Bin Wang, Zhijian Ou, and Zhiqiang Tan. Trans-dimensional random fields for language modeling. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, volume 1, pp. 785–794.