# An Adaptive Empirical Bayesian Method for Sparse Deep Learning

Wei Deng, Xiao Zhang, Faming Liang, Guang Lin

Purdue University, West Lafayette, IN, USA

October 23, 2019

# Overview

# Sampling in DNNs

Sampling in deep neural networks (DNN) has many desired properties.

- Asymptotic properties in modeling uncertainty, [9, 2].
- Proven guarantees in non-convex optimizations, e.g. [7, 11, 12, 6].

# Stochastic Gradient Langevin Dynamics

SGLD (no momentum) [10] is formulated as follows:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \epsilon^{(k)} \nabla_{\boldsymbol{\beta}} \tilde{L}(\boldsymbol{\beta}^{(k)}) + \mathcal{N}(0, 2\epsilon^{(k)}\tau^{-1}), \tag{1}$$

where $\nabla_{\boldsymbol{\beta}} \tilde{L}(\boldsymbol{\beta})$ is the stochastic gradient calculated from a mini-batch of data of size $n$ randomly sampled from the whole dataset of size $N$ to approximate the exact gradient $\nabla_{\boldsymbol{\beta}} L(\boldsymbol{\beta})$:

$$\nabla_{\boldsymbol{\beta}} \tilde{L}(\boldsymbol{\beta}) = \nabla_{\boldsymbol{\beta}} \log \mathrm{P}(\boldsymbol{\beta}) + \frac{N}{n} \sum_{i \in \mathcal{S}} \nabla_{\boldsymbol{\beta}} \log \mathrm{P}(\boldsymbol{d}_i | \boldsymbol{\beta}). \tag{2}$$

# Stochastic Gradient Hamiltonian Monte Carlo

SGHMC [3], proposes to generate samples as follows:

$$\begin{cases} d\boldsymbol{\beta} = \boldsymbol{r}dt, \\ \\ d\boldsymbol{r} = \nabla_{\boldsymbol{\beta}}\tilde{L}(\boldsymbol{\beta})dt - \boldsymbol{C}\boldsymbol{r}dt + \mathcal{N}(0, 2\boldsymbol{B}\tau^{-1}dt) + \mathcal{N}(0, 2(\boldsymbol{C} - \hat{\boldsymbol{B}})\tau^{-1}dt), \end{cases} \tag{3}$$

where $\boldsymbol{r}$ is the momentum item, $\hat{\boldsymbol{B}}$ is an estimate of the stochastic gradient variance, $\boldsymbol{C}$ is a user-specified friction term. Regarding the discretization of (3), we follow the numerical method proposed by [8] due to its convenience to import parameter settings from SGD.

# A class of Adaptive Stochastic Gradient MCMC

Unlike the exsiting framework for adaptive SG-MCMC [5], which aims to sample the original distribution $\pi_{\boldsymbol{\theta}}(\boldsymbol{\beta})$, our interest is to sample from **a new distribution** $\pi(\boldsymbol{\beta}, \boldsymbol{\theta}_*)$, where $\boldsymbol{\theta}_*$ is adaptively obtained by solving a fixed-point formulation $\int g_{\boldsymbol{\theta}_*}(\boldsymbol{\beta})\pi(\boldsymbol{\beta}, \boldsymbol{\theta}_*)d\boldsymbol{\beta} = \boldsymbol{\theta}_*$ and $g_{\boldsymbol{\theta}_*}$ here can be a closed-form expression update.

# A class of Adaptive Stochastic Gradient MCMC

The stochastic approximation algorithm can be used to solve the fixed-point iterations:

(1) Sample $\boldsymbol{\beta}^{(k+1)}$ from a transition kernel $\Pi_{\boldsymbol{\theta}^{(k)}}(\boldsymbol{\beta})$ based on SGMCMC, which yields the distribution $\pi(\boldsymbol{\beta}, \boldsymbol{\theta}^{(k)})$,

(2) Update
$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \omega^{(k+1)} H(\boldsymbol{\theta}^{(k)}, \boldsymbol{\beta}^{(k+1)}) = \boldsymbol{\theta}^{(k)} + \omega^{(k+1)}(h(\boldsymbol{\theta}^{(k)}) + \Omega^{(k)})$.

where $\omega^{(k+1)}$ is the step size. The equilibrium point $\boldsymbol{\theta}_*$ is obtained when the distribution of $\boldsymbol{\beta}$ converges to the invariant distribution $\pi(\boldsymbol{\beta}, \boldsymbol{\theta}_*)$.

# A class of Adaptive Stochastic Gradient MCMC

Stochastic approximation [1] differs from the Robbins-Monro algorithm in that sampling $\boldsymbol{\beta}$ from a transition kernel instead of a distribution introduces a Markov **state-dependent noise** $H(\theta_k, \boldsymbol{\beta}_{k+1}) - h(\theta_k)$, where $h(\boldsymbol{\theta})$ is the mean field function s.t. $h(\boldsymbol{\theta}) := \mathrm{E}[H(\boldsymbol{\beta}, \boldsymbol{\theta})]$.

# Convergence of Latent Variables

The key to guaranteeing the convergence of the adaptive SGLD algorithm is to use **Poisson's equation** to analyze additive functionals. By decomposing the Markov state-dependent noise $\Omega$ into martingale difference sequences and perturbations, where the latter can be controlled by the **regularity of the solution of Poisson's equation**, we can guarantee the consistency of the latent variable estimators.

## Theorem ($L_2$ convergence rate)

*For any $\alpha \in (0, 1]$, under assumptions in the appendix, the algorithm satisfies: there exists a constant $\lambda$ and a local optimum $\boldsymbol{\theta}^*$ such that*

$$\mathrm{E}\left[\|\boldsymbol{\theta}^{(k)} - \boldsymbol{\theta}^*\|^2\right] \leq \lambda \omega^{(k)},$$

# Weak Convergence of Samples

SGLD with adaptive latent variables forms a sequence of inhomogenous Markov chains and the weak convergence of $\boldsymbol{\beta}$ to the target posterior is equivalent to proving the weak convergence of SGLD with biased estimations of gradients. Inspired by [2], we have:

## Corollary

*Under assumptions in Appendix B.2, the random vector $\boldsymbol{\beta}^{(k)}$ from the adaptive transition kernel $\Pi_{\boldsymbol{\theta}^{(k-1)}}$ converges weakly to the invariant distribution $e^{\tau L(\boldsymbol{\beta},\boldsymbol{\theta}^*)}$ as $\epsilon \to 0$ and $k \to \infty$.*

# Application: A hierarchical formulation

We assume the weight $\beta_{lj}$ in sparse layer $l$ with index $j$ follow the spike-and-slab Gaussian-Laplace (SSGL) prior

$$\beta_{lj}|\sigma^2, \gamma_{lj} \sim (1-\gamma_{lj})\mathcal{L}(0, \sigma v_0) + \gamma_{lj}\mathcal{N}(0, \sigma^2 v_1).$$

The variance $\sigma^2$ follows an inverse gamma prior

$$\pi(\sigma^2) = IG(\nu/2, \nu\lambda/2).$$

The i.i.d. Bernoulli prior is used for $\boldsymbol{\gamma}$, namely

$$\pi(\boldsymbol{\gamma}_l|\delta_l) = \delta_l^{|\gamma_l|}(1-\delta_l)^{p_l-|\gamma_l|},$$

$\delta_l$ follows Beta distribution. The use of adaptive penalty enables to learn the level of sparsity automatically. Finally the posterior is as follows:

$$\pi(\boldsymbol{\beta}, \sigma^2, \delta, \boldsymbol{\gamma}|\mathcal{B}) \propto \pi(\mathcal{B}|\boldsymbol{\beta}, \sigma^2)^{\frac{N}{n}} \pi(\boldsymbol{\beta}|\sigma^2, \boldsymbol{\gamma})\pi(\sigma^2|\boldsymbol{\gamma})\pi(\boldsymbol{\gamma}|\delta)\pi(\delta)$$

# Sampling from Exact Likelihood with Empirical Prior

Instead of tackling $\pi(\boldsymbol{\beta}, \sigma^2, \delta, \boldsymbol{\gamma}|\mathcal{D})$ directly, we propose to iteratively update the expectation of the lower bound $Q$ by Fubini's theorem and Jensen's inequality:

$$\begin{aligned} & Q(\boldsymbol{\beta}, \sigma, \delta|\boldsymbol{\beta}^{(k)}, \sigma^{(k)}, \delta^{(k)}) \\ =& E_{\mathcal{B}}\left[E_{\boldsymbol{\gamma}|\mathcal{D}}\left[\log \pi(\boldsymbol{\beta}, \sigma^2, \delta, \boldsymbol{\gamma}|\mathcal{B})\right]\right]. \end{aligned}$$

Given $(\boldsymbol{\beta}^{(k)}, \sigma^{(k)}, \delta^{(k)})$ at the k-th iteration, we first sample $\boldsymbol{\beta}^{(k+1)}$ from $Q$, then optimize $Q$ with respect to $\sigma, \delta$ and $E_{\boldsymbol{\gamma}_l|\cdot,\mathcal{D}}$ via SA, where $E_{\boldsymbol{\gamma}_l|\cdot,\mathcal{D}}$ is used since $\boldsymbol{\gamma}$ is treated as unobserved variable.
We decompose our $Q$ as follows:

$$\begin{aligned} & Q(\boldsymbol{\beta}, \sigma, \delta|\boldsymbol{\beta}^{(k)}, \sigma^{(k)}, \delta^{(k)}) \\ =& Q_1(\beta, \sigma|\boldsymbol{\beta}^{(k)}, \sigma^{(k)}, \delta^{(k)}) \\ +& Q_2(\delta|\boldsymbol{\beta}^{(k)}, \sigma^{(k)}, \delta^{(k)}) + C, \end{aligned}$$

# Sampling from Exact Likelihood with Empirical Prior

$$Q_1(\beta|\beta^{(k)}, \sigma^{(k)}, \delta^{(k)})$$

$$= \overbrace{\frac{N}{n}\log \pi(\mathcal{B}|\beta)}^{\text{log likelihood}} - \overbrace{\sum_{l\in\mathcal{C}}\sum_{j\in p_l}\frac{\beta_{lj}^2}{2\sigma_0^2}}^{\text{non-sparse layers } \mathcal{C}} - \frac{p+\nu+2}{2}\log(\sigma^2)$$

$$- \overbrace{\sum_{l\in\mathcal{X}}\sum_{j\in p_l}\left[\frac{|\beta_{lj}|E_{\gamma_l|\cdot,\mathcal{D}}\left[\frac{1}{v_0(1-\gamma_{lj})}\right]}{\sigma}\right]}^{\text{deep SSGL priors in sparse layers } \mathcal{X}}$$

$$- \overbrace{\sum_{l\in\mathcal{X}}\sum_{j\in p_l}\left[\frac{\beta_{lj}^2 E_{\gamma_l|\cdot,\mathcal{D}}\left[\frac{1}{v_1\gamma_{lj}}\right]}{2\sigma^2}\right]}^{\text{deep SSGL priors in sparse layers } \mathcal{X}} - \frac{\nu\lambda}{2\sigma^2}$$

# Stochastic Approximation for Latent Variables

$$Q_2(\delta_l|\beta_l^{(k)}, \delta_l^{(k)}) = \sum_{l\in\mathcal{X}}\sum_{j\in p_l} \log\left(\frac{\delta_l}{1-\delta_l}\right) \overbrace{\mathop{\mathsf{E}}_{\gamma_l|\cdot,\mathcal{D}} \gamma_{lj}}^{\rho_{lj}} \tag{4}$$
$$+ (a-1)\log(\delta_l) + (p_l + b - 1)\log(1-\delta_l),$$

# Stochastic Approximation for Latent Variables

Regarding the closed-form updates with respect to $\boldsymbol{\rho}$, we denote the optimal $\boldsymbol{\rho}$ based on the current $\boldsymbol{\beta}$ and $\delta$ by $\tilde{\boldsymbol{\rho}}$. We have that $\tilde{\rho}_{lj}^{(k+1)}$, the probability of $\beta_{lj}$ being dominated by the $L_2$ penalty is

$$\tilde{\rho}_{lj}^{(k+1)} = \mathop{\mathrm{E}}_{\boldsymbol{\gamma}_l|\cdot,\mathcal{B}} \gamma_{lj} = \mathrm{P}(\gamma_{lj} = 1|\boldsymbol{\beta}_l^{(k)}, \delta_l^{(k)}) = \frac{a_{lj}}{a_{lj} + b_{lj}}, \tag{5}$$

where $a_{lj} = \pi(\beta_{lj}^{(k)}|\gamma_{lj} = 1)\delta_l^{(k)}$ and $b_{lj} = \pi(\beta_{lj}^{(k)}|\gamma_{lj} = 0)(1 - \delta_l^{(k)})$. Similarly, as to the updates w.r.t. $\boldsymbol{\kappa}$, the optimal $\tilde{\kappa}_{lj0}$ and $\tilde{\kappa}_{lj1}$ based on the current $\rho_{lj}$ are given by:

$$\tilde{\kappa}_{lj0} = \mathop{\mathrm{E}}_{\boldsymbol{\gamma}_l|\cdot,\mathcal{B}} \left[\frac{1}{v_0(1-\gamma_{lj})}\right] = \frac{1-\rho_{lj}}{v_0};\ \tilde{\kappa}_{lj1} = \mathop{\mathrm{E}}_{\boldsymbol{\gamma}_l|\cdot,\mathcal{B}} \left[\frac{1}{v_1\gamma_{lj}}\right] = \frac{\rho_{lj}}{v_1}. \tag{6}$$

# Stochastic Approximation for Latent Variables

To optimize $Q_1$ with respect to $\sigma$, by denoting $\text{diag}\{\kappa_{0li}\}_{i=1}^{p_l}$ as $\boldsymbol{\mathcal{V}_{0l}}$, $\text{diag}\{\kappa_{1li}\}_{i=1}^{p_l}$ as $\boldsymbol{\mathcal{V}_{1l}}$ we have:

$$\tilde{\sigma}^{(k+1)} = \begin{cases} \dfrac{R_b + \sqrt{R_b^2 + 4R_aR_c}}{2R_a} & \text{(regression)}, \\ \dfrac{C_b + \sqrt{C_b^2 + 4C_aC_c}}{2C_a} & \text{(classification)}, \end{cases} \tag{7}$$

where $R_a = N + \sum_{l\in\mathcal{X}} p_l + \nu$, $C_a = \sum_{l\in\mathcal{X}} p_l + \nu + 2$,
$R_b = C_b = \sum_{l\in\mathcal{X}} ||\boldsymbol{\mathcal{V}}_{0l}\boldsymbol{\beta}_l^{(k+1)}||_1$, $R_c = I + J + \nu\lambda$, $C_c = J + \nu\lambda$,
$I = \frac{N}{n}\sum_{i\in\mathcal{S}}\left(y_i - \psi(\boldsymbol{x}_i;\boldsymbol{\beta}^{(k+1)})\right)^2$, $J = \sum_{l\in\mathcal{X}} ||\boldsymbol{\mathcal{V}}_{1l}^{1/2}\boldsymbol{\beta}_l^{(k+1)}||^2$.[2]

[2]The quadratic equation has only one unique positive root. $\|\cdot\|$ refers to $L_2$ norm, $\|\cdot\|_1$ represents $L_1$ norm.

# Stochastic Approximation for Latent Variables

To optimize $Q_2$, a closed-form update can be derived from Eq.(4) and Eq.(5) given batch data $\mathcal{B}$:

$$\begin{aligned} \tilde{\delta}_l^{(k+1)} &= argmax_{\delta_l \in \mathbb{R}} Q_2(\delta_l | \beta_l^{(k)}, \delta_l^{(k)}) \\ &= \frac{\sum_{j=1}^{p_l} \rho_{lj} + a - 1}{a + b + p_l - 2}. \end{aligned} \tag{8}$$

# Pruning Strategy

Although the magnitude-based **unit pruning** shows more computational savings, it doesn't demonstrate robustness under coarser pruning. Pruning based on the probability $\rho$ is also popular in the Bayesian community, but achieving the target sparsity in sophisticated networks requires extra fine-tuning. We instead apply the magnitude-based **weight-pruning** to our compression experiments.

# Stochastic Approximatioin SGLD

**Algorithm 1** SGLD-SA with SSGL priors

**Initialize:** $\boldsymbol{\beta}^{(1)}$, $\boldsymbol{\rho}^{(1)}$, $\boldsymbol{\kappa}^{(1)}$, $\sigma^{(1)}$ and $\delta^{(1)}$ from scratch, set target sparse rates $\mathbb{D}$, $\mho$ and $\mathbb{S}$
**for** $k \leftarrow 1 : k_{\max}$ **do**
  **Sampling**
  $\boldsymbol{\beta}^{(k+1)} \leftarrow \boldsymbol{\beta}^{(k)} + \epsilon^{(k)} \nabla_{\boldsymbol{\beta}} Q(\cdot|\mathcal{B}^{(k)}) + \mathcal{N}(0, 2\epsilon^{(k)}\tau^{-1})$
  **Stochastic Approximation for Latent Variables**
  **SA**: $\boldsymbol{\rho}^{(k+1)} \leftarrow (1-\omega^{(k+1)})\boldsymbol{\rho}^{(k)} + \omega^{(k+1)}\tilde{\boldsymbol{\rho}}^{(k+1)}$ following Eq.(12)
  **SA**: $\boldsymbol{\kappa}^{(k+1)} \leftarrow (1-\omega^{(k+1)})\boldsymbol{\kappa}^{(k)} + \omega^{(k+1)}\tilde{\boldsymbol{\kappa}}^{(k+1)}$ following Eq.(13)
  **SA**: $\sigma^{(k+1)} \leftarrow (1-\omega^{(k+1)})\sigma^{(k)} + \omega^{(k+1)}\tilde{\sigma}^{(k+1)}$ following Eq.(14)
  **SA**: $\delta^{(k+1)} \leftarrow (1-\omega^{(k+1)})\delta^{(k)} + \omega^{(k+1)}\tilde{\delta}^{(k+1)}$ following Eq.(15)
  **if Pruning then**
    Prune the bottom-$s\%$ lowest magnitude weights
    Increase the sparse rate $s \leftarrow \mathbb{S}(1 - \mathbb{D}^{k/\mho})$
  **end if**
**end for**

# Simulation of Large-p-Small-n Regression

Dataset: $n = 100$ and $p = 1000$. $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\eta}$ where $\boldsymbol{X} \sim \mathcal{N}_p(0, \boldsymbol{\Sigma})$, $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, 0, 0, ..., 0)'$, $\boldsymbol{\eta} \sim \mathcal{N}_n(\boldsymbol{0}, 3\boldsymbol{I}_n)$, $\beta_1 \sim \mathcal{N}(3, \sigma_c^2)$, $\beta_2 \sim \mathcal{N}(2, \sigma_c^2)$, $\beta_3 \sim \mathcal{N}(1, \sigma_c^2)$, $\sigma_c = 0.2$.

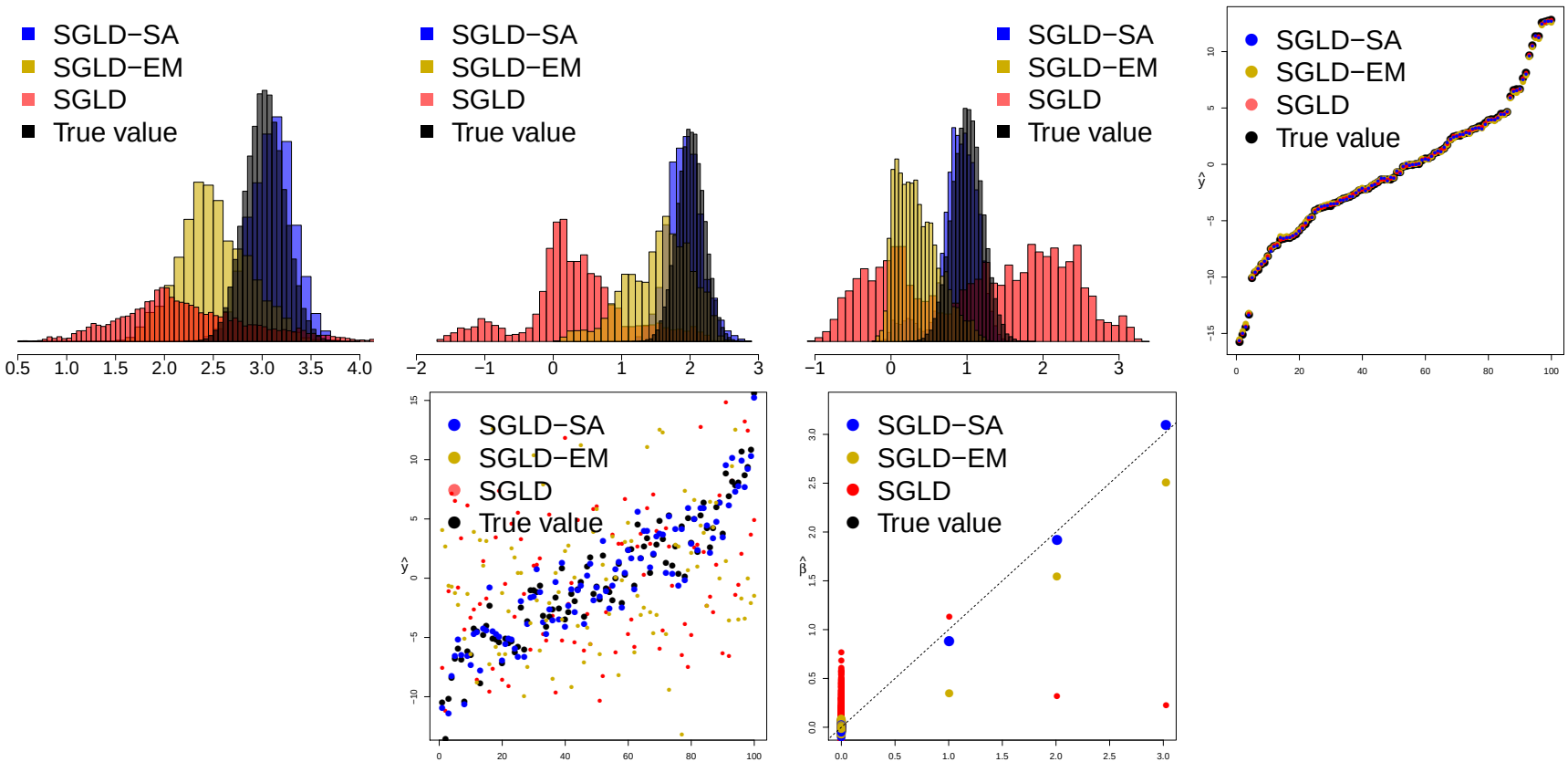



Figure: (a-c): Posterior estimation of $\beta_1$, $\beta_2$ and $\beta_3$, (d): training performance, (e): testing performance, (f): variable estimates

# Classification with Auto-tuning Hyperparameters

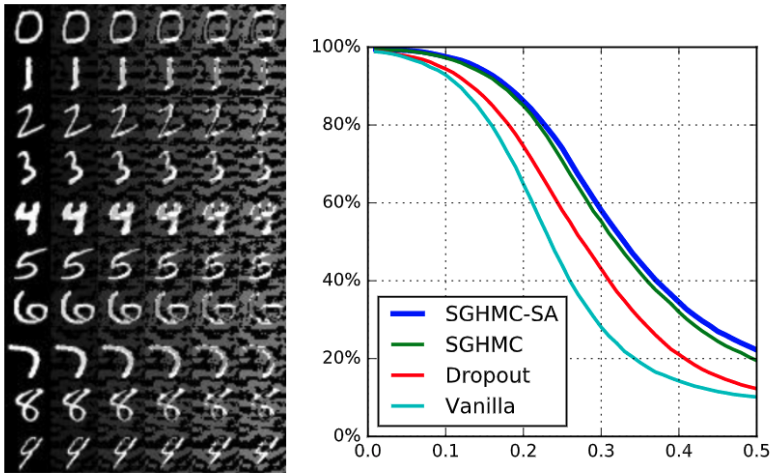
Fixed temperature can also be powerful in escaping "shallow" local traps [12], our temperatures are set to $\tau = 1000$ for MNIST and $\tau = 2500$ for FMNIST.

Table 2: Classification accuracy using shallow networks

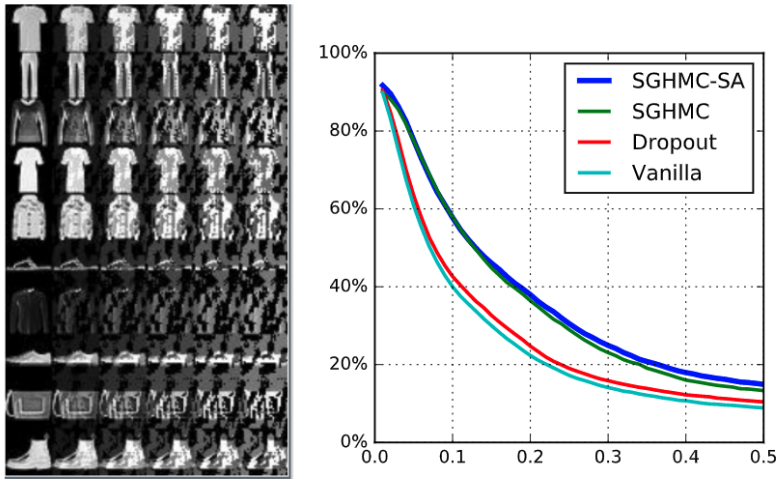
| DATASET | MNIST | DA-MNIST | FMNIST | DA-FMNIST |
|---|---|---|---|---|
| VANILLA | 99.31 | 99.54 | 92.73 | 93.14 |
| DROPOUT | 99.38 | 99.56 | 92.81 | 93.35 |
| SGHMC | 99.47 | 99.63 | 92.88 | 94.29 |
| **SGHMC-SA** | **99.59** | **99.75** | **93.01** | **94.38** |

Here, DA-MNIST and DA-FMNIST are tested with data augmentation and batch normalization, while MNIST and FMNIST are not.

# Defenses against Adversarial Attacks




(a) $\zeta = \ldots.$ (b) MNIST




(c) $\zeta = \ldots.$ (d) FMNIST

Figure 2: Adversarial test accuracies based on adversarial images of different levels

# Residual Network Compression

Table: Resnet20 Compression on CIFAR10. When $\mathbb{S} = 0.9$, we fix $v_0 = 0.005$, $v_1$ =1e-5; When $\mathbb{S} = 0.7$, we fix $v_0 = 0.1$, $v_1$ =5e-5; When $\mathbb{S} = 0.5$, we fix $v_0 = 0.1$, $v_1$ =5e-4; When $\mathbb{S} = 0.3$, we fix $v_0 = 0.5$, $v_1$ =1e-3.

| Methods \ $\mathbb{S}$ | 30% | 50% | 70% | 90% |
|---|---|---|---|---|
| A-SGHMC | 94.07 | 94.16 | 93.16 | 90.59 |
| A-SGHMC-EM | 94.18 | 94.19 | 93.41 | 91.12 |
| SGHMC-SA | 94.13 | 94.11 | 93.52 | 91.45 |
| **A-SGHMC-SA** | **94.23** | **94.27** | **93.74** | **91.68** |

Most notably, **91.68% accuracy based on 27K parameters (90% sparsity) in Resnet20 is the besting existing result**. By contrast, targeted dropout (2018) achieved 91.48% accuracy based on 47K parameters (90% sparsity) of Resnet32, BC-GHS (2017) achieved 91.0% accuracy based on 8M parameters (94.5% sparsity) of VGG models.

# Conclusion

In this paper, we propose a sparse Bayesian deep learning algorithm, SG-MCMC-SA, to adaptively learn the hierarchical Bayes mixture models in DNNs. This algorithm has four main contributions:

- We propose a novel AEB method to efficiently train hierarchical Bayesian mixture DNN models, where the parameters are learned through sampling while the priors are learned through optimization.
- We prove the convergence of this approach to the asymptotically correct distribution, and it can be further generalized to a general adaptive sampling algorithm for estimating state-space models in deep learning.
- We apply this adaptive sampling algorithm in the DNN compression problems firstly, with potential extension to a variety of model compression problems.
- It achieves the state of the art in terms of compression rates, which is 91.68% accuracy on CIFAR10 using only 27K parameters (90% sparsity) with Resnet20 [4].

# The End


A. Benveniste, M. Métivier, and P. Priouret, *Adaptive Algorithms and Stochastic Approximations*, Berlin: Springer, 1990.


C. Chen, N. Ding, and L. Carin, *On the Convergence of Stochastic Gradient MCMC Algorithms with High-order Integrators*, in Proc. of the Conference on Advances in Neural Information Processing Systems (NIPS), 2015, pp. 2278–2286.


T. Chen, E. B. Fox, and C. Guestrin, *Stochastic gradient Hamiltonian Monte Carlo*, in Proc. of the International Conference on Machine Learning (ICML), 2014.


K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.


Y.-A. Ma, T. Chen, and E. B. Fox, *A complete recipe for stochastic gradient MCMC*, in Proc. of the Conference on Advances in Neural Information Processing Systems (NIPS), 2015.


O. Mangoubi and N. K. Vishnoi, *Convex Optimization with Unbounded Nonconvex Oracles using Simulated Annealing*, in Proc. of Conference on Learning Theory (COLT), 2018.


M. Raginsky, A. Rakhlin, and M. Telgarsky, *Non-convex learning via Stochastic Gradient Langevin Dynamics: a nonasymptotic analysis*, in Proc. of Conference on Learning Theory (COLT), June 2017.


Y. Saatci and A. G. Wilson, *Bayesian GAN*, in Proc. of the Conference on Advances in Neural Information Processing Systems (NIPS), 2017, pp. 3622–3631.


Y. W. Teh, A. Thiéry, and S. Vollmer, *Consistency and Fluctuations for Stochastic Gradient Langevin Dynamics*, Journal of Machine Learning Research, 17 (2016), pp. 1–33.


M. Welling and Y. W. Teh, *Bayesian Learning via Stochastic Gradient Langevin Dynamics*, in Proc. of the International Conference on Machine Learning (ICML), 2011, pp. 681–688.


P. Xu, J. Chen, D. Zou, and Q. Gu, *Global Convergence of Langevin Dynamics Based Algorithms for Nonconvex Optimization*, in Proc. of the Conference on Advances in Neural Information Processing Systems (NIPS), Dec. 2018.


Y. Zhang, P. Liang, and M. Charikar, *A Hitting Time Analysis of Stochastic Gradient Langevin Dynamics*, in Proc. of Conference on Learning Theory (COLT), 2017, pp. 1980–2022.