3rd Symposium on Advances in Approximate Bayesian Inference, 2020 1–11

HWA: Hyperparameters Weight Averaging in Bayesian Neural Networks

Belhal Karimi BELHAL.KARIMI@GMAIL.COM
Cognitive Computing Lab, Baidu Research 10900 NE 8th St. Bellevue, WA 98004

Ping Li PINGLI98@GMAIL.COM
Cognitive Computing Lab, Baidu Research 10900 NE 8th St. Bellevue, WA 98004

Abstract

Bayesian neural networks attempt to combine the strong predictive performance of neural networks with formal quantification of uncertainty of the predicted output in the Bayesian framework. In deterministic deep neural network, the confidence of the model and the predictions at inference time are left alone. Applying randomness and Bayes Rule to the weights of a deep neural network is a step towards achieving this goal. Current state of the art optimization methods for training Bayesian Neural Networks are relatively slow and inefficient, compared to their deterministic counterparts. In this paper, we propose HWA (Hyperparameters Weight Averaging) algorithm that exploits an averaging procedure in order to optimize faster and achieve better accuracy. We develop our main algorithm using the simple averaging heuristic and demonstrate its effectiveness on the space of the hyperparameters of the networks random weights. Numerical applications are presented to confirm the empirical benefits of our method.

1. Introduction

While Deep Learning methods have shown increasing efficiency in various domains such as natural language processing, computer vision or robotics, sensible areas including autonomous driving or medical imaging not only require accurate predictions but also uncertainty quantification. In (Neal, 2012), authors develop a bayesian variant of plain feedforward multilayer neural networks in which weights and biases are considered as random variables. For supervised learning tasks, deterministic models are prone to overfitting and are not capable of estimating uncertainty in the training data which results in making overly confident decisions about the correct class, also known as *miscalibration* (Guo et al., 2017; Kendall and Gal, 2017). Nevertheless, representing that aforementioned uncertainty is crucial for decision making. Bayesian methods display a hierarchical probabilistic model that assume a (prior) random distribution over the parameters of the parameters and are useful for assessing the uncertainty of the model via posterior predictive distribution quantification (Blundell et al., 2015; Kingma et al., 2015). Current training methods for Bayesian Neural Networks (BNN) (Neal, 2012) include Variational Inference (Graves, 2011; Hoffman et al., 2013) or BayesByBackprop (Blundell et al., 2015) based on the Evidence Lower Bound (ELBO) maximization task. Naturally, Bayesian methods, and in particular BNNs, are highly sensitive to choice of the prior distribution parameters. Besides, current state-of-the-art models are not as efficient and robust as traditional deep learning models.

In this paper, we introduce a new *optimization* algorithm to alleviate those challenges. Our main contributions read as follows:

- We introduce Hyperparameter Weight Averaging (HWA), a training algorithm that leverages stochastic averaging techniques (Polyak and Juditsky, 1992) and posterior sampling methods to efficiently train bayesian neural networks.
- Given the high nonconvexity of the loss landscape, our method finds heuristic explanation from theoretical works on averaging and generalization such as (Keskar et al., 2016; He et al., 2019) and more practical work on Deep Neural Networks (DNN) optimization such as (Izmailov et al., 2018).
- We provide numerical examples showcasing the effectiveness of our method on simple and complex supervised classification tasks.

The remaining of the paper is organized as follows. Section 2 presents the related works in the fields of optimization, Variational Inference and posterior sampling. Section 3 introduces our main contribution, namely the HWA algorithm. Section 4 highlights the benefits of our procedure on various classification tasks. Section 5 concludes our work.

2. Related Work

Posterior Prediction. Due to the nonconvexity of the loss landscapes involved in modern deep learning tasks, sampling directly from the posterior distribution of the weights is not an option. Depending on the nature and the dimensionality of the problem, Markov Chain Monte Carlo (MCMC) methods have been employed to overcome this intractability issue. The samples drawn at convergence of the Markov chain are guaranteed to be drawn from the target distribution. Hamiltonian Monte Carlo (HMC) (Neal et al., 2011) or Metropolis Hastings (MH) (Hastings, 1970) are two standard solutions used in practice. Their stochastic gradients counterpart are extensively studied in (Ma et al., 2015).

Variational Inference (VI). When tackling an optimization problem, exact posterior sampling may be computationally involved and not even required. Variational inference was proposed in (Graves, 2011), in the particular case of BNNs, in order to fit a Gaussian variational posterior approximation over the weights of neural networks. Via a simple reparameterization trick (Blundell et al., 2015), several methods have emerged to train BNNs leveraging the ease of use and implementation of VI (Kingma et al., 2015; Blundell et al., 2015; Molchanov et al., 2017). Though, those methods appear to be inefficient for large datasets and newer ones were proposed to alleviate this issue such as normalizing flows (Louizos and Welling, 2017), deterministic VI (Wu et al., 2018) or dropout VI (Gal and Ghahramani, 2016).

Stochastic Averaging. Averaging methods include the seminal papers of (Polyak, 1990) and (Ruppert, 1988), both based on the combination of past iterates along a stochastic approximation trajectory. For nonconvex objectives, this averaging procedure has been adapted to Stochastic Gradient Descent (SGD) trajectory in (Zhou and Cong, 2017). In particular, for recent deep learning examples, Izmailov et al. (2018) develops a method that averages snapshots of a DNN through the iterations leading to a better empirical generalization. Those experimental discoveries are then backed by theoretical understanding of the multilayer neural network loss landscape in (Keskar et al., 2016; He et al., 2019).

3. Hyperparameters Averaging in Bayesian Neural Networks

In this section, we introduce the basic concepts of bayesian neural networks and their associated loss function which plays a key role in this paper. From an optimization perspective, we first review the Stochastic Weight Averaging (SWA) (Izmailov et al., 2018) procedure, which can be seen as an approximation of the mean trajectory of the SGD iterates and then introduce our method – namely HWA. *While SWA averages snapshots of the weights of the neural networks from successive past iterations, our method HWA only captures snapshots of the hyperparameters, and not of the weights that are sampled at each training iteration.* We then discuss the uncertainty estimation prediction of such method and how our proposed extra step, combining *posterior sampling* and *optimization*, can lead to better generalization of the trained model on test sets. Based on the idea of ensemble learning, as in (Garipov et al., 2018), averaging procedure leads to a similar ensemble effect. Indeed, popular Bayesian methods for training BNNs tend to focus on a single mode, leading in general to mode collapse, whereas, as stated and exhaustively tested in (Fort et al., 2019), ensembles tend to explore diverse modes in function space, so are SWA and HWA.

3.1. Bayesian Neural Networks and ELBO Maximization

Let $((x_i, y_i), i \in [1, n])$ be i.i.d. input-output pairs and $w \in \mathcal{W} \subseteq \mathbb{R}^d$ be a latent variable. When conditioned on the input data $x = (x_i, i \in [n])$, the joint distribution of $y = (y_i, i \in [n])$ and w is given by:

$$p(y, w|x) = \pi(w) \prod_{i=1}^{n} p(y_i|x_i, w) \ . \tag{1}$$

In the particular case of BNN, this likelihood function is parameterized by a multilayer neural network, which can be convolutional or not. The latent variables w are thus the weights and the biases of the model and are considered as latent (and random) variables. Training such hierarchical models involves sampling from the posterior distribution of the weights w conditioned on the data (x, y), noted $p(w|y, x)$. In most cases, this posterior distribution $p(w|y, x)$ is intractable and is approximated using a family of parametric distributions, $\{q(w, \boldsymbol{\theta}), \boldsymbol{\theta} \in \Theta\}$. The variational inference (VI) problem (Blei et al., 2017) boils down to minimizing the Kullback-Leibler (KL) divergence between $q(w, \boldsymbol{\theta})$ and the posterior distribution $p(w|y, x)$. The objective is the ELBO (Evidence Lower BOund) and reads:

$$\mathcal{L}(\boldsymbol{\theta}) := -\mathbb{E}_{q(w;\boldsymbol{\theta})}\big[\log p(y|x, w)\big] + \mathbb{E}_{q(w;\boldsymbol{\theta})}\big[\log q(w; \boldsymbol{\theta})/\pi(w)\big] \ . \tag{2}$$

Directly optimizing the objective function in (2) can be difficult. First, with $n \gg 1$, evaluating the objective function $\mathcal{L}(\boldsymbol{\theta})$ requires a full pass over the entire dataset. Second, for some complex models, the expectations in (2) can be intractable even if we assume a simple parametric model for $q(w; \boldsymbol{\theta})$. Thus, the computation of the gradient requires an approximation step usually invoking a Monte Carlo (MC) approximation step.

Training solutions simply include using SGD (Bottou and Bousquet, 2008) where the gradient of the individual ELBO (2) is computed using Automatic Differentiation (Kucukelbir et al., 2017). The final update goes in the opposite direction of that gradient up to a learning rate factor. In the sequel, we develop an improvement over baseline SGD, invoking averaging virtue of several successive snapshots of the gradients. The method, called Hyperparameters Weight Averaging (HWA), aims at improving the generalization property of the trained model on unseen data.

3.2. Averaging model snapshots through hyperparameters loss landscapes

Consider a deterministic deep neural network, the idea behind the Stochastic Weight Averaging (SWA) procedure, developed in (Izmailov et al., 2018), is to run several iterates of the classical SGD procedure, starting from a pre-trained model. At each timestep noted T_{avg}, the model estimate is set to be the the average of the last T_{avg} iterates. After establishing the connectivity between several modes (point estimates of minimal loss) of the same deep neural network (after different training procedures) in (Garipov et al., 2018), the ability to average over all those iterates probably traversing several models, or at least model estimates that belong to low loss region, would make the resulting trained model more robust and thus generalize better to unseen data. Several theoretical papers such as (He et al., 2019) or (Keskar et al., 2016) provide justifications of this empirical phenomena.

– Hyperparameters Weight Averaging: Based on the probabilistic model developed Section 3.1, the loss function (2) is defined on the space of the hyperparameters, *i.e.* the mean and the variance of the variational candidate distribution. Regarding the parameterization choice of the variational candidate $q(w; \boldsymbol{\theta})$, we choose for simplicity a scalar mean μ_ℓ depending on the layer $\ell \in [1, L]$ and constant between each neuron of the same layer. Classically, the covariance of this variational distribution is diagonal, see (Kirkpatrick et al., 2017; Blundell et al., 2015), yet this assumption can be too restrictive. We follow the direction taken in (Maddox et al., 2019), where the covariance of $q(w, \boldsymbol{\theta})$ is a diagonal matrix. As a result, the averaging procedure practically occurs on the set of hyperparameters and requires updating the mean and the variance of the variational candidate distribution, at iteration $k+1$, if k, the iteration index, is a multiple of the cycle length c, as below:

$$\mu_\ell^{HWA} = \frac{n_{\mathrm{m}} \mu_\ell^{HWA} + \mu_\ell^{k+1}}{n_{\mathrm{m}} + 1} \quad \text{and} \quad \sigma^{HWA} = \frac{n_{\mathrm{m}} \sigma^{HWA} + (\mu_\ell^{k+1})^2}{n_{\mathrm{m}} + 1} - (\mu_\ell^{HWA})^2 \,, \tag{3}$$

where for all $\ell \in [1, L]$, μ_ℓ^{k+1} and σ^{k+1} are obtained via Stochastic VI (Hoffman et al., 2013).

Algorithm 1 HWA: Hyperparameters Weight Averaging

1: **Input:** Iteration index k. Trained hyperparameters $\hat{\mu}_\ell$ and $\hat{\sigma}$. LR γ_k. Cycle length c. Gradient vector $\nabla \mathcal{L}_{i_k}(\theta^k)$
2: $\gamma \leftarrow \gamma(k)$ (Cyclical LR for the iteration)
3: **SVI updates:**
4: $\quad \mu_\ell^{k+1} \leftarrow \mu_\ell^k - \gamma_k \nabla_{\mu_\ell} \mathcal{L}_{i_k}(\mu_\ell^k)$
5: $\quad \sigma^{k+1} \leftarrow \sigma^k - \gamma_k \nabla_\sigma \mathcal{L}_{i_k}(\sigma^k)$
6: **if** $\mathrm{mod}(k, c) = 0$ **then**
7: $\quad n_\mathrm{m} \leftarrow k/c \quad$ (Number of models to average over)

$$\mu_\ell^{HWA} \leftarrow \frac{n_\mathrm{m} \mu_\ell^{HWA} + \mu_\ell^{k+1}}{n_\mathrm{m} + 1} \quad \text{and} \quad \sigma^{HWA} \leftarrow \frac{n_\mathrm{m} \sigma^{HWA} + (\mu_\ell^{k+1})^2}{n_\mathrm{m} + 1} - (\mu_\ell^{HWA})^2$$

8: **end if**
9: **Return** hyperparameters $(\{\mu_\ell^{HWA}\}_{l=1}^L, \sigma^{HWA})$.

Our main contribution lies in Algorithm 1. Stochastic Variational updates are executed Line 4. The stochastic averaging procedure happens every c iterations, and consists in computing the weighted sum between the latest model estimate and the running average noted by the superscript HWA. Once the parameter estimates are updated via (3), the network weights are then *sampled* according to the updated variational candidate distributions in order to compute the next iteration approximate stochastic gradient, see Alg. 2 in supplementary material for more details on the end-to-end VI procedure embedding HWA.

Note that in the above procedure, the variational candidate $q(w, \boldsymbol{\theta})$ has a diagonal covariance matrix where the scalar standard deviations are obtained through Algorithm 1. Yet, it is also possible build a non diagonal proposal covariance to bypass the restriction of such structure. Besides, given the nonconvexity and high dimensionality of the true posterior distribution, adding a *low rank non diagonal* structure to the covariance of our proposal would yield a gain in efficiency in the VI procedure. Of course the ideal option would be to construct a curvature-informed covariance for our proposal but at a higher cost. The low-rank plus diagonal posterior approximation matrix, noted Σ of $q(w, \boldsymbol{\theta})$ introduced in (Maddox et al., 2019) reads:

$$\Sigma = \frac{1}{2} \Sigma_\mathrm{diag} + \frac{\widehat{D} \widehat{D}^\top}{2(R-1)} \tag{4}$$

where $\mu^{HWA} = (\mu_\ell^{HWA}, \ell \in [1, L])$, R is the maximum number of columns in the low rank deviation matrix $\widehat{D}$ and Σ_diag is the diagonal covariance defined above. The r-th component of the low rank deviation matrix $\widehat{D}$ is defined as the gap between the current estimate and the running average: $\widehat{D}_r = \theta_r - \theta_r^{HWA}$. It quantifies how far the current estimate parameter deviate from the current average. Several hyperparameters are worth highlighting here. The standard learning rate γ_k plays a key role and is either a constant or cyclical, see (Zhang et al., 2019). The cycle length c, monitoring the number of times snapshots of the model estimates are being averaged, is also of utmost importance and needs careful tuning.

3.3. Comparison with other classical averaging procedures in nonconvex optimization

From Alg. 1, we note that the averaging procedure happens once at each cycle c, a tuning hyperparameter, on the parameter estimates resulting from a simple stochastic gradient descent update. Yet, another natural averaging step would be to keep K snapshots of the past stochastic gradients and compute an aggregated sum used as the drift term in the general update rule, see (Zhou and Cong, 2017). Nevertheless, in our setting, the objective function while being nonconvex is (possibly) parameterized by a high dimensional neural network making it computationally involved to store those K gradients.

Incremental Aggregated Gradients methods: Popular optimization methods, such as SAG (Schmidt et al., 2017) or SAGA (Defazio et al., 2014), make use of the past individual gradient and compute a moving average of those vectors as the final drift term. Those methods are proven to be faster than plain SGD in both convex and nonconvex cases, leveraging among other reasons variance reduction effect, but suffer from a high storage cost. Indeed the drift term is composed of the sum of the n past individual gradient where n is equal to the size of the training set.

MISO (Mairal, 2015): Another important method invoking variance reduction through incremental update of the drift term in a gradient descent step is the Minimization by Incremental Surrogate Optimization method, namely MISO, developed in Mairal (2015) (see (Karimi et al., 2019) for its doubly stochastic variant, relevant in this setting). Contrary to the method mentionned above, the accumulation does not happen on the gradient but on the sum of individual surrogate objective functions. While this framework is more general than SAG or SAGA, and also does not require storing n past gradients, it is still computationally heavy to store those n past objective functions, rather their model parameter estimates, when tackling deep neural networks training.

For all those reasons, HWA surely combines the virtue of the accumulation/aggregation effect and the low computing cost of vanilla SGD.

3.4. Embedding HWA in Variational Inference

The end to end VI method embedding our HWA procedure in summarized in Alg. 2. During the traditional VI routine, the covariance of the proposal $q(\cdot)$ is either set to (3) or (4).

Discussion on the choice of the variational candidate distribution: The general aim of the update rules presented above is to construct an efficient variational candidate distribution that would provide an approximate shape of the true posterior. Our method acts on the mean and covariance of a simple Gaussian distribution where the covariance matrix is either diagonal or low rank. Nevertheless, other choice of proposal can be employed such as the spike and slab variational distribution, in (Gal and Ghahramani, 2016), leveraging dropout mechanism in VI. The other similar idea, namely concrete dropout in (Gal et al., 2017) not only optimizes the hyperparameters of the weights but also the dropout probabilities. We do not consider those variants as our work focuses on Gaussian approximations of the posterior distribution and how their parameters are updated, see Section 4 for a description of baseline methods used in our numerical experiments.

We now give in Alg. 2, the overall training algorithm of the bayesian neural network using the proposed HWA algorithm to update the parameters.

Algorithm 2 Variational Inference with HWA for BNNs

1: **Input:** Trained hyperparameters $\hat{\mu}_\ell$ and $\hat{\sigma}$. Sequence of LR $\{\gamma_k\}_{k>0}$. Cycle length c. K iterations.
2: **for** $k = 0, 1, ...$ **do**
3: Sample an index i_k uniformly on $[n]$
4: Sample MC batch of weights $\{w_k^m\}_{m=1}^{M_k}$ from variational candidate $q(w, \theta^k)$ with $\theta^k = (\mu^k, \Sigma^k)$ and the covariance is either diagonal (3) or low rank (4).
5: Compute MC approximation of the gradient vectors:

$$\nabla \mathcal{L}_{i_k}(\theta^k) \approx \frac{1}{M_k} \sum_{m=1}^{M_k} \log p(y_{i_k} | x_{i_k}, w_m^k) + \nabla KL(q(w, \theta^k) || \pi(w))$$

6: Update the vector of parameter estimates calling Alg. 1: $(\mu^K, \Sigma^K) = \mathrm{HWA}(k, c, \gamma_k, \nabla \mathcal{L}_{i_k}(\theta^k))$
7: **end for**
8: **Return** Fitted parameters (μ^K, Σ^K).

4. Numerical Experiments

We provide experiments on classification tasks with various neural network architectures and datasets to demonstrate the effectiveness of our method, namely HWA.

Methods. We consider three baselines: vanilla BayesByBackprop (BBB) developed in (Blundell et al., 2015), the Stochastic Gradient Langevin Dynamics (SGLD) method introduced in (Welling and Teh, 2011) and its cyclical variant in (Zhang et al., 2019). The algorithms are initialized at the same point and the results are averaged over 5 repetitions.

Datasets. We compare the different algorithms on *MNIST* (LeCun, 1998) and *CIFAR10* (Krizhevsky and Hinton, 2009) datasets.

Network architectures. (MNIST) We train a Bayesian variant of LeNet-5 convolutional neural network (LeCun et al., 1998) on the MNIST dataset. Under the prior distribution π, see (1), the weights are assumed independent and identically distributed according to $\mathcal{N}(0, 1)$. We also assume a Gaussian variational candidate distribution such that $q(\cdot; \boldsymbol{\theta}) \equiv \mathcal{N}(\mu, \sigma^2 \mathbf{I})$, where $\mathbf{I}$ is the identity matrix. The variational posterior parameters are thus $\boldsymbol{\theta} = (\mu, \sigma)$ where $\mu = (\mu_\ell, \ell \in [d])$ with d the number of weights in the neural network. (CIFAR-10) We train the Bayesian variant of the VGG neural network (Simonyan and Zisserman, 2014) on the CIFAR-10 dataset. As in the previous example, the weights are assumed i.i.d. according to $\mathcal{N}(0, \mathbf{I})$. Standard hyperparameters values found in the literature, such as the annealing constant or the number of MC samples, were used for the benchmark methods. For better efficiency and lower variance, the Flipout estimator (Wen et al., 2018) is used to compute the MC approximation of the gradient of the loss function.

Results. Results for both datasets and network architectures are reported Figure 1. While for the MNIST dataset, the runs for HWA and SGLD are comparable both in terms of train and testing loss and accuracy, they both highlights better convergence properties compared to BAYESBYBACKPROP (BBB). It is worth mentioning that our method HWA displays a similar behavior as a gradient based method, such as SGLD, by only leveraging the average of past snapshots of the variational candidate hyperparameters. Regarding the CIFAR10 experiment, our method shows the lowest training loss and generalize better to unseen data (cf. last figure on bottom line in Figure 1). In conclusion, HWA achieves state-of-the-art results for either small or large bayesian variants of standard network architectures while using a simple and efficient averaging update at each *cycle*.

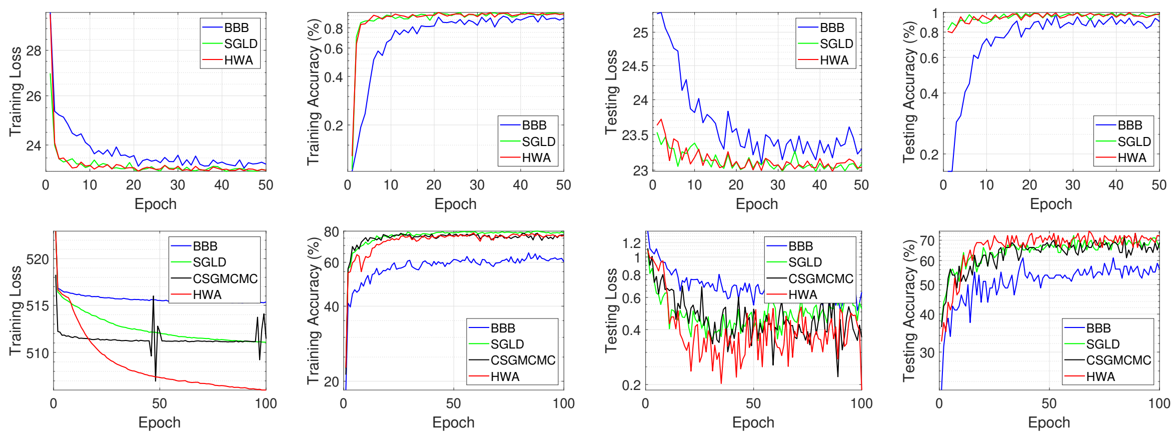

Figure 1: Comparison for Bayesian LeNet CNN architecture on MNIST dataset (top) and Bayesian VGG architecture on CIFAR-10 dataset (bottom). The plots are averaged over 5 repetitions.

5. Conclusion

We present in this paper an averaging procedure on the hyperparameters of the weights of a bayesian neural network architecture. Based on both empirical and theoretical results regarding stochastic averaging, we propose the HWA algorithm in order to increase the generalization ability of a BNN. The procedure is easily implementable on top of any vanilla optimizer with standard design choices for prior and candidate distributions, crucial quantities in variational inference. Numerical experiments show the advantage of our method matching and sometimes surpassing baselines such as SGLD or CSGMCMC, which require additional expensive gradient computation.

References

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. In *Advances in neural information processing systems*, pages 3581–3590, 2017.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, pages 8789–8798, 2018.

Alex Graves. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356, 2011.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*, 2017.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 04 1970. ISSN 0006-3444. doi: 10.1093/biomet/57.1.97. URL https://doi.org/10.1093/biomet/57.1.97.

Haowei He, Gao Huang, and Yang Yuan. Asymmetric valleys: Beyond sharp and flat local minima. In *Advances in Neural Information Processing Systems*, pages 2553–2564, 2019.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

Belhal Karimi, Hoi-To Wai, and Eric Moulines. A doubly stochastic surrogate optimization scheme for non-convex finite-sum problems. *Submitted paper*, 2019.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, pages 2575–2583, 2015.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational bayesian neural networks. *arXiv preprint arXiv:1703.01961*, 2017.

Yi-An Ma, Tianqi Chen, and Emily Fox. A complete recipe for stochastic gradient mcmc. In *Advances in Neural Information Processing Systems*, pages 2917–2925, 2015.

Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164, 2019.

Julien Mairal. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational dropout sparsifies deep neural networks. *arXiv preprint arXiv:1701.05369*, 2017.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Boris T Polyak. A new method of stochastic approximation type. *Avtomat. i Telemekh*, 7: 98:107, 1990.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

David Ruppert. Efficient estimations from a slowly convergent robbins-monro process. Technical report, Cornell University Operations Research and Industrial Engineering, 1988.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv:1803.04386*, 2018.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E Turner, José Miguel Hernández-Lobato, and Alexander L Gaunt. Deterministic variational inference for robust bayesian neural networks. *arXiv preprint arXiv:1810.03958*, 2018.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. *arXiv preprint arXiv:1902.03932*, 2019.

Fan Zhou and Guojing Cong. On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*, 2017.