entropy

MDPI

Article

Stochastic Control for Bayesian Neural Network Training

Ludwig Winkler [1,*], César Ojeda [2] and Manfred Opper [2,3]

[1] Machine Learning Group, Technische Universität Berlin, 10623 Berlin, Germany
[2] Artificial Intelligence Group, Technische Universität Berlin, 10623 Berlin, Germany
[3] Centre for Systems Modelling and Quantitative Biomedicine, University of Birmingham, Birmingham B15 2TT, UK
* Correspondence: winkler@tu-berlin.de

Abstract: In this paper, we propose to leverage the Bayesian uncertainty information encoded in parameter distributions to inform the learning procedure for Bayesian models. We derive a first principle stochastic differential equation for the training dynamics of the mean and uncertainty parameter in the variational distributions. On the basis of the derived Bayesian stochastic differential equation, we apply the methodology of stochastic optimal control on the variational parameters to obtain individually controlled learning rates. We show that the resulting optimizer, StochControlSGD, is significantly more robust to large learning rates and can adaptively and individually control the learning rates of the variational parameters. The evolution of the control suggests separate and distinct dynamical behaviours in the training regimes for the mean and uncertainty parameters in Bayesian neural networks.

Keywords: Bayesian inference; Bayesian neural networks; learning

Citation: Winkler, L.; Ojeda, C.; Opper, M. Stochastic Control for Bayesian Neural Network Training. *Entropy* **2021**, *24*, 1097. https://doi.org/10.3390/e24081097

Academic Editor: Patrick Shafto

Received: 4 July 2022
Accepted: 30 July 2022
Published: 9 August 2022

1. Introduction

Deep Bayesian neural networks (BNNs) aim to leverage the advantages of two different methodologies. First, in recent years, deep representations have been incredibly successful in fields as diverse as computer vision, speech recognition and natural language processing [1–3]. Much of the success, however, revolves around prediction accuracy. Second, Bayesian methodologies are required to obtain an estimate of model uncertainty, a crucial feature that allows deep neural networks to tackle risk assessment to create informed model decisions. The role of model uncertainty in the training procedure of BNNs, however, remains unaddressed; the present investigation seeks to exploit the model uncertainty in Bayesian neural networks for the development of new learning algorithms.

For the training of BNNs, the approximate posterior over the model parameters is obtained via a maximization of the variational lower bound.

Such a posterior introduces a form of uncertainty in the parameters which is different than that injected by random batches of data. In this investigation, we seek to exploit both the data uncertainty (aleatoric) and the model uncertainty (epistemic) to solve a control problem aimed at maximizing the evidence lower bound (ELBO), where the control parameters gauge the dynamics of the gradient during descent.

The contributions of our work are threefold,

- We provide a derivation of the stochastic differential equation on a first principle basis that governs the evolution of the parameters in variational distributions trained with variational inference and we decompose the uncertainty of the gradients into their aleatoric and epistemic components.
- We derive a stochastic optimal control optimization algorithm which incorporates the uncertainty in the gradients to optimally control the learning rates for each variational parameter.

- The evolution of the control exhibits distinct dynamical behaviour and demonstrates different fluctuation and dissipation regimes for the variational mean and uncertainty parameters.

Section 1 offers an introduction to the topic. In Section 2, we provide an overview over probabilistic models and Bayesian neural networks. Section 3 details the derivation of the stochastic differential equation governing the dynamics of the frequentist and variational parameters. Subsequently, we derive a stochastic optimal control algorithm in Section 4 on the basis of the dynamics of the variational parameters. Finally, Section 5 summarizes experiments undertaken and the performance of the stochastic optimal control optimizer, as well as the distinct behaviour of the control parameters.

2. Variational Inference for Bayesian Neural Networks

For a training dataset $\mathcal{D}$, the Bayesian formulation of a neural network places a posterior distribution $p(\theta|\mathcal{D})$ and a prior $p(\theta)$ on each of its parameters θ. The quintessential task in Bayesian inference is to compute the posterior $p(\theta|\mathcal{D})$ according to Bayes' rule:

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}. \tag{1}$$

Given a likelihood function $p(\mathcal{D}|\theta)$ and the parameter prior $p(\theta)$, we can make predictions by marginalizing out over the parameters. For the most common application of supervised learning with label y and data x, $\mathcal{D} = \{y_m, x_m\}_{m=1}^N, x \in \mathcal{X}, y \in \mathcal{Y}$, this gives us

$$p(y|x) = \int p(y|x,\theta)p(\theta|\mathcal{D})d\theta. \tag{2}$$

where $p(y|x,\theta)$ is the likelihood function of the output y given the input x and the posterior parameter distribution $p(\theta|\mathcal{D})$. For highly parameterized models, the inference of the posterior distribution $p(\theta|\mathcal{D})$ requires the computation of a high dimensional integral which is numerically intractable to compute for most complex models as they can easily have millions of parameters.

There are two main approaches for inferring the posterior distribution $p(\theta|\mathcal{D})$ in Bayesian neural networks: sampling from the posterior distribution in proportion to the data likelihood and prior [4], and variational inference, which optimizes a bound on the evidence and approximates the true posterior with a tractable distribution $q(\theta|\phi) \approx p(\theta|\mathcal{D})$ with the variational parameters ϕ, [5].

Our approach focuses on the variational inference formulation, which scales well to large data regimes as the bound is amenable to gradient-based optimization schemes [6]. Variational inference infers the posterior distribution by optimizing the Kullback–Leibler divergence between the true posterior $p(\theta|\mathcal{D})$ and a variational distribution $q(\theta|\phi)$. The important detail is that the variational distribution is assumed to be independent of the data $\mathcal{D}$, which makes the solution approximate yet tractable. The optimization problem is then

$$\arg\min_{\phi} \mathbb{KL}[q(\theta|\phi)||p(\theta|\mathcal{D})] = \arg\min_{\phi} -\mathbb{E}_{q(\theta|\phi)}[\log p(\mathcal{D}|\theta)] + \mathbb{KL}[q(\theta|\phi)||p(\theta)]. \tag{3}$$

We are, thus, left to optimize the ELBO, which is derived in full in Appendix A, in the form $\mathbb{E}_{q(\theta|\phi)}[\log p(\mathcal{D}|\theta)] - \mathbb{KL}[q(\theta|\phi)||p(\theta)]$ as a surrogate loss function. The ELBO is optimized numerically through gradient descent algorithms, which bring their own set of challenges with respective to gradient step size, directional sensitivity and exploding and vanishing gradients. We propose a stochastic optimal control algorithm for gradient descent optimization which controls the learning rate for every variational parameter ϕ based on the local surface of the Kullback–Leibler divergence.

For the remainder of this paper, we assume that the variational distribution $q(\theta|\phi)$ for each parameter θ follows an independent normal or Laplace distribution with the location of the distribution μ and the scale σ as the variational parameters $\phi = \{\mu, \sigma\}$ of the parameter θ. Since the scale parameter σ is constrained to be positive, we employ an additional reparameterization $\sigma = \log(\exp(\rho) + 1)$ which allows us to compute derivatives for ρ during optimization while keeping σ strictly positive.

2.1. Stochastic Differential Equations for Frequentist Models

During optimization, the model parameters follow a dynamical process; in the following section, we show how it is possible to approximate this dynamic as an SDE; we start with a frequentist version where no distribution is imposed on the parameters (as in the BNN), and the stochasticity is injected by the dataset and samples from therein. Given a probabilistic model $p(y_m|x_m, \theta) : \mathcal{X} \to \mathcal{Y}$ with a set of scalar parameters $\theta \in \mathbb{R}$, the input $x_m \in \mathcal{X}$ and output $y_m \in \mathcal{Y}$, we compute the derivative of a scalar loss function $\mathcal{L}_m : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ to obtain the derivative with respect to each parameter $\partial_\theta \mathcal{L}_m$ in the probabilistic model for a single data point. Gradient descent requires us to calculate the derivative of the loss over the entire training dataset $\mathcal{D}$ at each iteration $\partial_\theta \mathcal{L} = 1/|\mathcal{D}| \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}_i$. This gradient, has an associated variance:

$$\sigma_{\mathcal{D}} = \mathbb{V}_D[\partial_\theta \mathcal{L}] = \frac{1}{|\mathcal{D}|} \sum_{i \in \mathcal{D}} (\partial \mathcal{L}_i - \partial_\theta \mathcal{L})^2. \tag{4}$$

The computational cost of calculating gradients over entire training datasets is prohibitively expensive, which has favoured the use of mini-batch sampled gradients. Now, a mini-batch with $M \ll |\mathcal{D}|$ data points is sampled [7]. The assumption is that a mini-batch is computationally tractable while providing a representative sample of the training dataset to compute a sufficiently good gradient on. We denote $\mathcal{D}_m$ as a single data sample and $\mathcal{D}_M$ as the mini-batch sample. The sampling of the mini-batches introduces stochasticity into the gradient estimation. The first and second moments, denoted as $\mathbb{E}[\cdot]$ and $\mathbb{V}[\cdot]$, for each scalar parameter θ of the mini-batch gradients are:

$$\partial_\theta \mathcal{L}_M = \frac{1}{M} \sum_{m=1}^{M} \partial_\theta \mathcal{L}_m \tag{5}$$

$$\partial_\theta \mathcal{L} = \mathbb{E}_{M \sim p(D)}[\partial_\theta \mathcal{L}_M] \tag{6}$$

$$\mathbb{V}_M[\partial_\theta \mathcal{L}_M] = \frac{1}{M} \sum_{i \in \mathcal{D}} (\partial \mathcal{L}_i - \partial_\theta \mathcal{L})^2 \tag{7}$$

$$\mathbb{V}_M[\partial_\theta \mathcal{L}_M] \propto \frac{1}{M} \sigma_{\mathcal{D}} \tag{8}$$

It is easy to see that we can decrease the variance in the gradient estimation by increasing the size of the mini-batch M. The change in the parameters $\Delta\theta_t = \theta_{t+1} - \theta_t$ in gradient based optimization consequentially follows a noisy estimate of the true gradient $\partial_\theta \mathcal{L}$ which is distributed according to the first- and second-order moments in (4) and (5). The central limit theorem implies that the derivatives are distributed along a Gaussian distribution, $\partial_\theta \mathcal{L}_M \sim \mathcal{N}(\partial_\theta \mathcal{L}, \frac{1}{M}\sigma_{\mathcal{D}})$ [8]. Given the distribution of the gradients, the evolution of the parameter through time with the learning rate η can be approximated by:

$$\theta_{t+1} = \theta_t - \eta \partial_\theta \mathcal{L}_M \tag{9}$$

$$\Delta\theta_t = -\eta \partial_\theta \mathcal{L} + \eta \sqrt{\frac{\sigma_{\mathcal{D}}}{M}}\, \epsilon \quad ; \epsilon \sim \mathcal{N}(0, 1) \tag{10}$$

This formulation of the parameter dynamics during training has strong similarities with the Euler–Maruyama discretization of an Ito drift–diffusion process. Indeeed, for an SDE with drift $b(\theta_t)$ and diffusion $\sigma(\theta)$:

$$d\theta_t = b(\theta_t)dt + \sigma(\theta_t)dW_t \tag{11}$$

we have the associated Eurler–Maruyama discretization:

$$\theta_{t+1} = \theta_t + b(\theta_t)\Delta t\sqrt{\Delta t}\sigma(\theta)\epsilon. \tag{12}$$

We proceed by setting $\eta \equiv \Delta t$, $b(\theta_t) \equiv \partial_\theta \mathcal{L}$ and $\sigma(\theta_t) \equiv \sqrt{\eta / M\sigma_{\mathcal{D}}}$ to denote equivalency, as further described in [8–10]. This modification allows the use of stochastic analysis to Ito drift-diffusion processes. See [11] for a more thorough discussion on the relationship of the learning rate and the diffusion of SGD). If we additionally consider the learning in the infinitesimal limit of $\eta \to 0$, we arrive at a formulation for the instantaneous change in time which is given by

$$d\theta_t = -\partial_\theta \mathcal{L}dt + \sqrt{\eta / M\sigma_{\mathcal{D}}}dW_t \tag{13}$$

which is a stochastic differential equation, where dW_t is a Wiener process that originates from the limit applied to $\sqrt{\eta}\epsilon$ [12]. We can, thus, conclude that the change in the parameters θ_t, for an infinitesimal small learning rate η, follows a stochastic differential equation in the form of an Ito drift–diffusion process over time in which the sampling of the mini-batches contributes the diffusion [12].

2.2. Stochastic Differential Equations for Bayesian Models

In BNN models, each scalar parameter θ is modelled by a univariate distribution $\theta \sim q(\theta|\phi)$. The use of the distribution $q(\theta|\phi)$ extends the loss $\mathcal{L}$ to the form of the ELBO which is additive in the mini-batch samples m and has a closed form regularization term (the Kullback–Leibler divergence between posterior $p(\theta|\mathcal{D})$ and prior $p(\theta)$), the derivation of which can be found in Appendix A. Not only do we choose data samples at random, but, concurrently, we sample the parameter θ from the distribution $q(\theta|\phi)$ following the reparametrization trick. The parameter θ is thus a random variable itself. Consequentially, the derivative $\partial_\theta \mathcal{L}_m$ for a single data sample m will exhibit randomness originating both from the randomly sampled mini-batches and the stochasticity of the sampled parameters from the variational distribution.

The uncertainty of the parameter derivative $\partial_\theta \mathcal{L}$ can be decomposed into the aleatoric and the epistemic uncertainty. The aleatoric uncertainty arises from the variance in the data and is irreducible, whereas the epistemic uncertainty arises from the uncertainty of the parameter θ and can be reduced to zero, since, in principle, the parameters can be sampled θ.

Employing the tractable univariate variational distribution $q(\theta|\phi)$ to achieve a scalable optimization, for a derivative $\partial_\theta \mathcal{L}_m$ which is dependent on the random parameter θ and the randomly chosen data sample m, we can decompose the uncertainty of $\partial_\theta \mathcal{L}$ into a sum of the data uncertainty and the parameter uncertainty, which follows from the law of total variance [13]:

$$\mathbb{V}[\partial_\theta \mathcal{L}] = \underbrace{\mathbb{V}_{p(\mathcal{D}_M)}\Big[\mathbb{E}_{q(\theta|\phi)}[\partial_\theta \mathcal{L}_m|\mathcal{D}_m]\Big]}_{\text{Aleatoric Uncertainty}} + \underbrace{\mathbb{E}_{p(\mathcal{D}_M)}\Big[\mathbb{V}_{q(\theta|\phi)}[\partial_\theta \mathcal{L}_m|\mathcal{D}_m]\Big]}_{\text{Epistemic Uncertainty}}. \tag{14}$$

In effect, we draw samples twice in BNNs to obtain 'per data sample per variational sample' derivatives: data samples for the mini-batch and parameter samples θ from the variational distribution. Aleatoric uncertainty first computes the expectancy over the 'variationally sampled' derivatives per data sample $\mathcal{D}_m$ and subsequently computes variance over the mini-batch $\mathcal{D}_M$. Epistemic uncertainty first computes the variance over the 'variationally sampled' gradients and, finally, computes the expected derivative over the mini-batch $\mathcal{D}_M$)

It is important over which source of randomness the variance is computed in the uncertainty decomposition. The first term, $\mathbb{V}[\mathbb{E}[\partial_\theta \mathcal{L}_m|\mathcal{D}_m]]$, represents the aleatoric uncertainty and measures the data uncertainty. It measures how much the average gradient varies over the dataset. The second term, $\mathbb{E}[\mathbb{V}[\partial_\theta \mathcal{L}_m|\mathcal{D}_m]]$, is called the epistemic uncertainty and measures the uncertainty originating from the model parameter distribution. For the epistemic uncertainty, the variance is computed over the source of parameter uncertainty and averaged over the data samples. In BNNs this is explicitly modelled through the use of distributions for every parameter θ. Frequentist models exhibit only aleatoric uncertainty, as the variance over the deterministic gradients in the epistemic uncertainty evaluates to zero.

For a univariate variational distribution $\theta \sim q(\theta|\phi)$, we can now formulate the stochastic differential equation (SDE) that governs the dynamics of the variational parameters $\phi = \{\mu, \sigma\}$.

The first modification, with respect to the SDE of a frequentist model in Equation (10) is that, for every parameter θ in the frequentist model, we have, in fact, two separate variational parameters $\phi = \{\mu, \sigma\}$ in the Bayesian model, corresponding to the mean and scale of the variational distribution from which we sample θ. We, thus, have the two differential equations for the variational parameters $\{\mu, \sigma\}$,

$$d\mu_t = -\mathbb{E}\big[\partial_\mu \mathcal{L}\big]dt + \mathbb{V}\big[\partial_\mu \mathcal{L}\big]^{\frac{1}{2}} dW_t \tag{15}$$

$$d\sigma_t = -\mathbb{E}[\partial_\sigma \mathcal{L}]dt + \mathbb{V}[\partial_\sigma \mathcal{L}]^{\frac{1}{2}} dW_t^*. \tag{16}$$

in which $d\sigma_t$ has a separate Wiener process dW_t^* due to the externalized noise in the reparameterization, the details of which can be checked up upon in the Appendix C. The second modification is the separation of uncertainty, given that we have the additional source of uncertainty from the distribution $q(\theta|\phi)$. We can, thus, employ the uncertainty decomposition to obtain

$$d\mu_t = -\mathbb{E}\big[\partial_\mu \mathcal{L}\big]dt + \Big(\mathbb{V}\big[\mathbb{E}\big[\partial_\mu \mathcal{L}_m|\mathcal{D}_m\big]\big] + \mathbb{E}\big[\mathbb{V}\big[\partial_\mu \mathcal{L}_m|\mathcal{D}_m\big]\big]\Big)^{\frac{1}{2}} dW_t \tag{17}$$

$$d\sigma_t = -\mathbb{E}[\partial_\sigma \mathcal{L}]dt + \Big(\mathbb{V}[\mathbb{E}[\partial_\sigma \mathcal{L}_m|\mathcal{D}_m]] + \mathbb{E}[\mathbb{V}[\partial_\sigma \mathcal{L}_m|\mathcal{D}_m]]\Big)^{\frac{1}{2}} dW_t^* \tag{18}$$

We can now see that the only difference in the SDEs that govern the training dynamics in frequentist and Bayesian models is the added epistemic uncertainty in the diffusion term of the Bayesian stochastic differential equation.

Figure 1 exemplifies the different terms in the Bayesian stochastic differential equation and how uncertainty in stochastic gradient descent for a variational distribution can be decomposed for a toy example in one dimension. The details of the derivation of the Bayesian stochastic differential equation can be followed up in Appendix C.

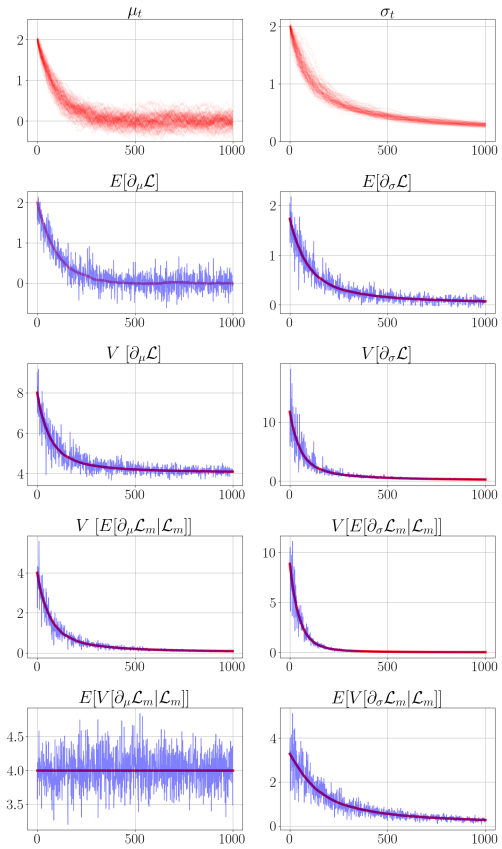

Figure 1. The components of the stochastic differential equation for the variational parameters μ_t and σ_t over time. The empirical drift and diffusion estimates shown in blue are unbiased estimates of the true analytically derived drift and diffusion terms. The loss was $\mathcal{L} = \frac{1}{2}(\theta_t - b)^2$ where b was sampled randomly from $b \in \{-2, +2\}$ to simulate aleatoric uncertainty. The aleatoric uncertainty from the data in the gradients remains constant whereas the epistemic uncertainty from the parameter distribution is reduced to zero.

3. Stochastic Control for Learning Rates

Having derived and characterized the training dynamics of the variational parameters $x \equiv \{\mu_t, \sigma_t\}$ on a first principle basis, we now construct our proposed stochastic optimal control algorithm for BNNs. Our approximation methodology relies on the limit $\eta \to 0$. We first introduce a new control variable that respects the limit, namely the learning rate adjustment to the training, an additional adaptive diagonal control matrix U, which leads to a full SDE for the dynamics of training as:

$$\dot{x}_t = -U\, \mathbb{E}[\nabla_x \mathcal{L}]\, dt + U\sqrt{\eta}\mathbb{V}[\nabla_x \mathcal{L}]^{\frac{1}{2}} dW_t \tag{19}$$

which is an Ito drift-diffusion process, where both the drift and the diffusion are controlled by the diagonal control matrix U and where the diffusion term is estimated from the variance of the gradients. We essentially scale it on a per-parameter basis with the control matrix U. We clip the individual control parameters U_i on the diagonal of U to the range $U_i \in [0, 1]$ bounding the step size to $\eta U_i \in [0, \eta]$. We posed our problem as follows: if we have the gradients $\nabla_x \mathcal{L}$, how do we choose the policy for adjusting the control parameter U to minimize the loss at the end of the training? Essentially:

$$\min_u \mathbb{E}[\mathcal{L}(X_T)] \text{ subject to (19)} \tag{20}$$

provided that X follows Equation (19). The general optimal control formalism requires us to minimize the cost C for the optimal control parameter U, accumulated over time $t \in [t_0, t_1]$, and the final cost $C(x_t, U, t_1)$, under the constraint of the dynamics $\dot{x}_t$.

3.1. Simplifiying the Loss

It is known that the loss surface $\mathcal{L}$ of deep neural network architectures is highly non-linear, which makes global optimization nearly impossible. In a similar way to [14,15], we therefore approximate the loss surface locally with a quadratic function of the form

$$g(x_t) = \frac{1}{2}(x_t - b)^\top A(x_t - b). \tag{21}$$

The quadratic approximation as seen in Figure 2 forfeits the global loss surface for a local approximation in which the respective optimal quantities can be computed optimally in the sense of the local approximation. This simplification is chosen such that a tractable stochastic optimal control algorithm can be derived. Intuitively, given a local quadratic approximation of the loss surface, the offset parameter b denotes the optimum of the quadratic approximation $g(x_t)$, whereas the curvature A denotes how flat or steep the loss surface is locally.

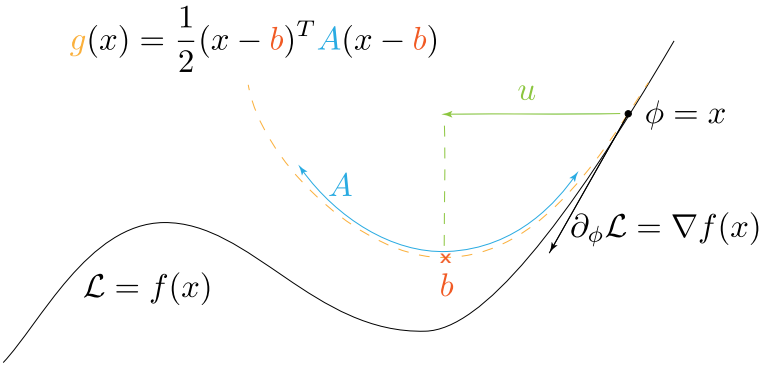

Figure 2. A one-dimensional illustration of how the optimal stochastic control u is determined from the gradient and parameter information. The parameters ϕ and their gradient information $\partial_\phi \mathcal{L}$ are used to estimate the curvature A and offset b for the quadratic approximation g through which the optimal control parameter u is determined. In our experiments with Bayesian neural networks, each parameter θ has two variational parameters $\phi = \{\mu, \sigma\}$, such that $A \in \mathbb{R}^{2x2}$ and $b \in \mathbb{R}^2$.

Consequently, we want to move the variational parameters $\{\mu_t, \rho_t\}$ in the observable state vector x_t as close as possible to this local optimum which coincides with the offset parameter b.

The curvature A and the offset parameter b of the local quadratic approximation of the loss surface can be conveniently calculated via ordinary least squares with the gradient relation (see Appendix D for details)

$$\nabla_\phi \mathcal{L} \sim \nabla_x g(x_t) = A(x_t - b). \tag{22}$$

We maintain running averages of the gradients and the parameters to prevent abrupt changes in the control. The quadratic approximation of the loss surface is maintained for each parameter distribution in the BNN architectures.

3.2. Our Control Problem

Taking inspiration from the local quadratic approximation $g(x_t)$, we wish to minimize the distance of the observable state variables x_t to the optimum b of the quadratic approximation $g(x)$. We introduce an auxilliary variable L which allows us to simplify the classical

control problem that requires the solution for the Hamiltonian–Jacobi–Bellman equation Appendix E.

It is known, by definition, that $M_{ij} = (x_i - b_i)(x_j - b_j)$ is a stochastic variable. We can obtain a relationship between L and the approximation of the error g:

$$g(x_t) = \frac{1}{2} M_{11} A_{11} + M_{12} A_{12} + \frac{1}{2} M_{22} A_{22} \tag{23}$$

We make use of Ito's lemma, detailed in Appendix B, to obtain the dynamics of the error dM and define the diffusion matrix $D = \eta \mathbb{V}[\nabla_{x_t} \mathcal{L}]$ which gives us,

$$dM = \left(-\nabla_x M \, U \, \nabla_x \mathcal{L} + \frac{\eta}{2} \mathrm{Tr}[D U^T \, \nabla_x^2 M \, U] \right) dt + \nabla_x M^T U \sqrt{\eta \mathbb{V}[\nabla_x \mathcal{L}]} dW$$

which is again an Ito drift-diffusion process, and for which we provide the relevant gradient calculations in Appendix D. With the intention of separating the drift and evaluating the matrix derivatives, the details of which are in the Appendix D, we obtain

$$f(M, U, t) = -(UAM + MAU) + \eta \, UDU \tag{24}$$

The dynamics of the error $f(M, U, t)$ denote the drift of the Ito drift-diffusion process dM and represent the average dynamics of the error function over time, given the dynamics dx_t of the parameters $x_t = (\mu_t, \rho_t)^\top$. The task which we want to achieve is to minimize the loss in (23) in such a way that we arrive at the optimum after the control period $t \in [t_0, t_1]$.

$$C(M, U, t_1) = \frac{1}{2} \int_{t_0}^{t_1} \mathrm{Tr}\left[A \dot{M}\right] dt. \tag{25}$$

where A is the curvature of the local approximation $g(x_t)$. The motivation of this formulation is that M measures the distance of the state vector x_t to the local optimum b in the quadratic approximation scaled by the curvature A. Thus, minimizing the distance M at each time step is equivalent to minimizing the entire cost C. The full derivation can be found in Appendix E.

The optimization of the final cost C can be solved by minimizing the cost of $\mathrm{Tr}[A\dot{M}]$, which, in turn, minimizes C. Taking the derivative of $\frac{1}{2}\mathrm{Tr}[A\dot{M}]$ with respect to the individual control parameters U_{ii} and setting it to zero gives us

$$U' = (A \circ D)^{-1} \frac{\mathrm{Diag}[AMA]}{\eta} \tag{26}$$

where $U' = [U_{11}, U_{22}]^\top$ is a vector with the corresponding control parameters, $\circ$ is the Hadamard product and Diag$[\cdot]$ extracts the diagonal elements of a matrix For indefinite matrices A, we project U' onto the eigenvector corresponding to the positive eigenvalue to ensure that the optimality condition is met [16]. The full derivation can be reviewed in Appendix E.

We compute the control parameter U' jointly for the variational parameters $\{\mu, \sigma\}$, which results in the matrices A, D, M to be in $\mathbb{R}^{2\times 2}$. The inversion of the 2×2 matrices can be performed analytically, as detailed at the end of Appendix F. Comparing the operations required per parameter in ADAM (addition, subtraction, division etc.) and those in StochControlSGD (mostly 2×2 matrix multiplications and analytical inversions), we arrive at an approximately $2.5\times$ increase in computations for StochControlSGD compared to ADAM. It is important to note that ADAM has to be applied to both variational parameters independently, whereas StochControlSGD computes the control parameters jointly, thus saving computation.

The StochControlSGD algorithm is detailed in its entirety in Algorithm 1.

Algorithm 1: StochControlSGD

Result: Optimal stochastic control parameter U

Variational parameters $\phi = \{\mu, \rho\}$;

for *Minibatch B* **do**

  Compute per sample gradients $\nabla_\phi \mathcal{L}_m$;

  Update running average for first order moments of $\nabla_\phi \mathcal{L}_m$ and ϕ ;

  Compute OLS parameters A,b, and D, M ;

  Compute U from A, D and M ;

  $\phi \leftarrow \phi - \eta U \nabla_\phi \mathcal{L}_B$;

end

4. Experiments

We evaluate the proposed stochastic optimal control SGD, which we abbreviate as StochControlSGD, on the MNIST [17], FashionMNIST [18] and CIFAR10 [19] datasets. In Table 1, we compare the final performance of StochControlSGD with the performance of ADAM, controlled SGD (cSGD), SGD and SGD with cosine learning rate scheduling, as proposed by [15].

Table 1. Test accuracy on the MNIST, FMNIST and CIFAR10 datasets. We abbreviate StochControlSGD as scSGD, and the SGD with cosine learning rate scheduling as LRSGD, for notational brevity. The best performing optimization algorithm per data set is denoted in bold.

	MNIST					**FMNIST**					**CIFAR10**				
	SGD	**ADAM**	**cSGD**	**scSGD**	**LRSGD**	**SGD**	**ADAM**	**cSGD**	**scSGD**	**LRSGD**	**SGD**	**ADAM**	**cSGD**	**scSGD**	**LRSGD**
NN	0.959	**0.987**	0.961	/	0.985	0.818	**0.890**	0.851	/	0.878	0.461	**0.512**	0.432	/	0.499
CNN	0.989	**0.993**	0.981	/	0.990	0.904	**0.918**	0.912	/	0.907	0.853	**0.865**	0.857	/	0.855
BNN (Normal)	0.956	0.963	0.970	**0.971**	0.069	0.865	0.870	0.876	**0.900**	0.900	0.441	0.442	0.451	**0.471**	0.462
CBNN (Normal)	0.982	0.988	0.982	**0.990**	0.989	0.869	0.914	0.903	**0.921**	0.915	0.615	**0.854**	0.836	0.853	0.801
BNN (Laplace)	0.976	**0.978**	0.974	0.977	0.975	0.890	0.875	**0.903**	0.901	0.9	**0.501**	0.452	0.461	0.479	0.500
CBNN (Laplace)	0.989	0.987	0.985	**0.991**	0.989	0.899	0.916	0.907	**0.918**	0.912	0.627	0.857	0.829	**0.857**	0.853

Learning rate scheduling was chosen as the cosine annealing, where the initial learning rate was chosen as 10^{-1} and was decreased to 10^{-5}. The experimental setup is detailed in Appendix G.

ADAM provides a strong baseline for the frequentist models when the learning rate is chosen to be appropriately small. Following the notion of learning rate scheduling, we initialized the learning of both cSGD and StochControlSGD as $\eta = 0.5$ and $u_0 = 1.0$. Both cSGD and StochControlSGD are able to adaptively and individually set their control parameters over the course of optimization.

Additionally, we plot the convergence of the ADAM, cSGD and StochControlSGD in Figure 3. The results are portrayed more concisely in Figure 4, for which five runs for each learning rate and each optimizer are combined in a boxplot format.

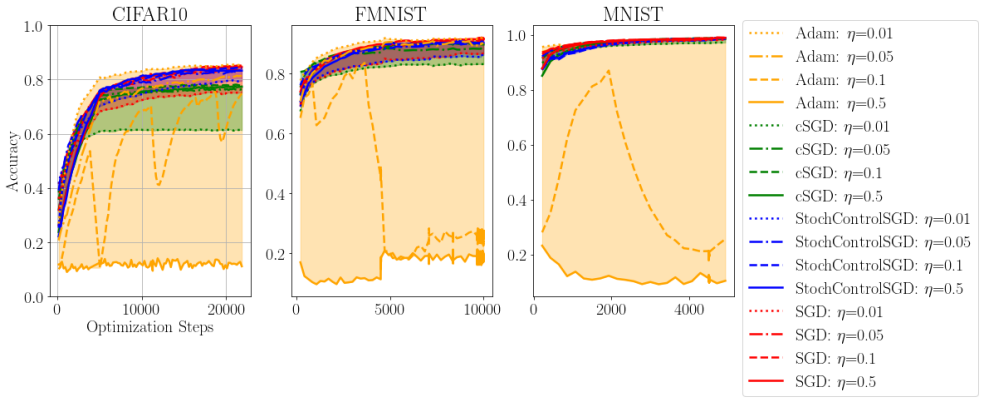

Figure 3. Comparison of StochControlSGD with SGD, controlled SGD and ADAM. StochcontrolSGD offers very robust performance over varying learning rates.

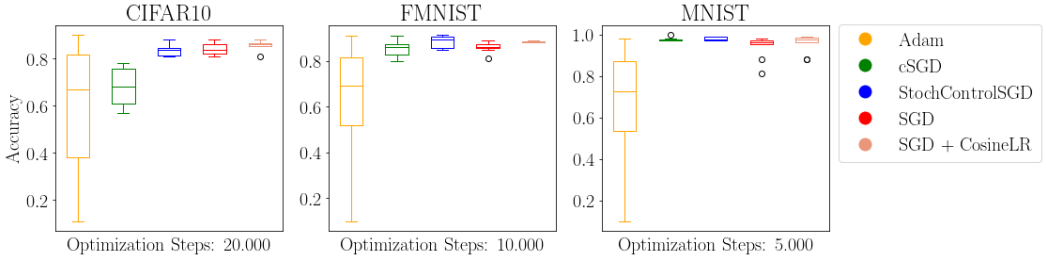

Figure 4. Combined performance of the optimizers over different learning rates. StochControlSGD provides reliable performance over a wide range of learning rates without the necessity of hyperparameter tuning.

In contrast to cSGD and StochControlSGD, ADAM does not have the ability to modify the a priori chosen learning rate η. Coupled with the first- and second-order moments from which the surrogate gradient is computed, ADAM is sensitive to the large learning rate with significantly worsening performance for learning rates at $\eta = 0.5$ and $\eta = 0.1$. The larger learning rates do not pose a problem for the optimal control optimizers cSGD and StochControlSGD, as they can adaptively and individually control their learning rates. We consider only optimizers which rely on the gradient information to accelerate the gradient descent and forego learning rate scheduling algorithms which incorporate performance information, such as learning rate schedulers which decrease the learning rate if a performance plateau is detected.

Among the optimal control optimizers, StochControlSGD provides tighter bounds on the lower and upper performance while offering a higher performance. Especially on the CIFAR10 dataset in Figure 3, StochControlSGD improves upon cSGD with better absolute performance and less variation between the largest learning rate of $\eta = 0.5$ and the smallest learning rate $\eta = 0.01$. Furthermore, it can be seen that the performance of StochControlSGD and cSGD improve with larger learning rates. As can be seen in Figure 3,

the performance of the largest learning rate of $\eta = 0.5$ is, in fact, its best performance, whereas it is the worst performance for ADAM.

The direct comparison of ADAM with StochControlSGD connects to recent work carried out by [20] on the fundamental optimization of deep Bayesian models with gradient optimization algorithms developed for frequentist models. The methodology of BNNs is limited in the amount of relevant information in the uncertainty with respect to the learning optimization due to its reliance on normal priors. Modern frequentist deep neural networks rely on custom layer architectures, such as BatchNorm [21], with additional data augmentation schemes, which have no clear Bayesian interpretation, raising additional questions on the applicability of porting frequentist ideas, such as layer designs, in deep neural networks, to their Bayesian formulations.

Behaviour of Control Parameter

The evolution of the control parameter U allows insight into the descent and fluctuation behaviour of the variational parameters μ_t and ρ_t with respect to the ELBO. More specifically, it allows us to shed some light onto the dynamics between the data log likelihood and the KL divergence.

The data loglikelihood aims at minimizing the uncertainty parameter ρ_t of each variational distribution as much as possible. The gradients of KL divergence, in turn, prioritize an uncertainty parameter which corresponds to the prior which we chose as $\mathcal{N}(0, I)$. The relative weighting of the data log likelihood and KL divergence with respect to the number of samples in the ELBO heavily favours the gradients of the data log likelihood during the descent phase for large datasets. As the gradients of the KL divergence are independent of the data by definition, the importance of their gradients increases proportionally to the diminishing gradients of the converging data log likelihood.

The uncertainty parameters were initialized to $\rho_0 = -6.9$ in all our experiments which allows the BNN to increase the uncertainty of select parameters if the KL divergence dominates the gradients of the specific parameter in question. The intuition is that deep neural networks, in fact, only use few weights [22], and, thus, the uncertainty parameters can be maximized by the KL divergence for parameters for which the gradients of the KL divergence are stronger than the gradients originating from the data log likelihood.

We can observe this behaviour in Figure 5, where the median control parameters of μ_t decrease quickly alongside the control parameters for the uncertainty parameter ρ_t. However, as the data loglikelihood converges, the median control parameter of the uncertainty parameter is increased as the relative importance of the gradients originating from the data loglikelihood decreases and the gradients from the KL divergence dominate.

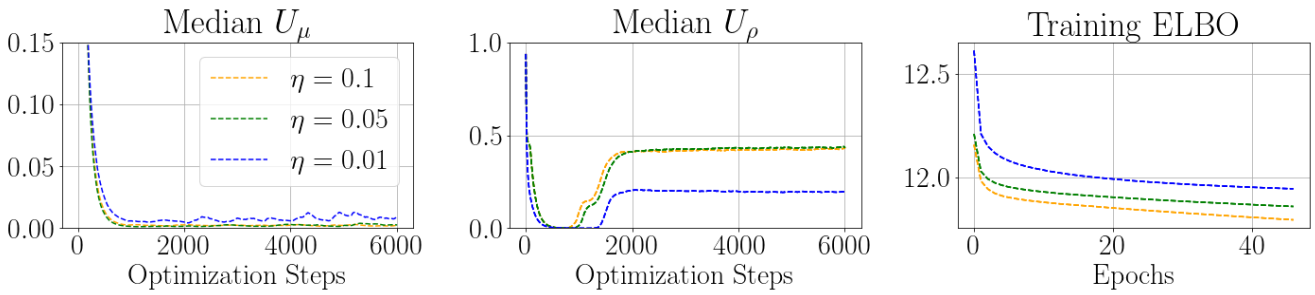

Figure 5. The median control parameter over time plotted with the Training ELBO which is used to compute the gradients for a BNN which was trained on Fashion MNIST.

This indicates two different dynamical regimes in the optimization of the uncertainty parameter of the variational distribution. The mean control parameter remains small during the descent and fluctuation dynamics whereas the uncertainty control is, in fact, increased by the stochastic control optimization algorithm in the fluctuation phase.

5. Related Work

The authors of [15] derived an optimal control algorithm for frequentist models which incorporated the variance into the learning rate scheduling. In [23], it was argued that instead of decreasing the learning in the dissipation phase of the optimization, the batch size should be increased to reduce the uncertainty in the gradients. The authors of [24] and [25] examined adaptive learning rate schemes for changing loss surfaces. The idea of a priori cyclical scaling in the learning rates was pioneered in [26].

The use of the reparameterization of the Gaussian variational distribution in deep Bayesian neural networks to arrive at a scalable optimization algorithm based on variational inference was proposed in [27]. The authors of [28] examined the behaviour of DropOut [29] as an approximate Bayesian inference. The authors of [30] demonstrated that the dropout rate could be learned as an approximate uncertainty parameter.

6. Conclusions

We have examined the potential for incorporating Bayesian uncertainty information directly into a learning algorithm. For this, we derived the SDEs for variational parameters on a first principle basis. With both aleatoric and epistemic uncertainty present in the optimization process, we decomposed the diffusion parameter of the SDE into its data and parameter uncertainties.

Having identified the underlying dynamics of the variational parameters during optimization, we proceeded to formulate a stochastic optimal control algorithm for Bayesian models which was able to incorporate the Bayesian uncertainty information into an adaptive and selective learning rate schedule. An analysis of the control parameters indicated separate dynamical behaviours during optimization of the mean and uncertainty parameters. This can be investigated further to examine the dynamics of the ELBO as a loss function for other probabilistic models.

Author Contributions: Conceptualization, C.O. and M.O.; methodology, L.W. and C.O.; software, L.W.; validation, L.W., C.O. and M.O.; formal analysis, L.W.; investigation, L.W.; writing—original draft preparation, L.W. and C.O.; writing—review and editing, M.O.; visualization, L.W. All authors have read and agreed to the published version of the manuscript.

Funding: The research of M.O was partially funded by the Deutsche Forschungsgemeinschaft (DFG)—Project-ID 318763901—SFB1294. The research of L.W. and C.O. was funded by the BIFOLD-Berlin Institute for the Foundations of Learning and Data (ref. 01IS18025A and ref 01IS18037A).

Data Availability Statement: The training data and code base used in this study are available upon reasonable request from the authors.

Acknowledgments: Ludwig Winkler would like to thank Klaus-Robert Müller for fruitful discussions and proof reading and Jason Salomon Rinnert for technical support. We acknowledge support by the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Conflicts of Interest: The authors declare no conflict of interest.The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A. Evidence Lower Bound

In variational inference for Bayesian neural networks, we want to minimize the Kullback–Leibler divergence between the true posterior distribution $p(\theta|\mathcal{D})$ and the variational distribution $q(\theta|\phi)$, which is easy to work with and from which we can sample easily:

$$\arg\min_{\phi} \; \mathbb{KL}[q(\theta|\phi)||p(\theta|\mathcal{D})] \tag{A1}$$

$$= \arg\min_{\phi} \; \mathbb{E}_{q(\theta|\phi)}\left[\log \frac{q(\theta|\phi)}{p(\theta|\mathcal{D})}\right] \geq 0. \tag{A2}$$

The posterior distribution $p(\theta|\mathcal{D})$ conditioned on the data $\mathcal{D}$ can be rewritten according to the Bayes theorem as

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})} \tag{A3}$$

with which we can rewrite the Kullback–Leibler divergence as

$$\mathbb{E}_{q(\theta|\mathcal{D})}\left[\log \frac{q(\theta|\phi)}{p(\theta|\mathcal{D})}\right] \tag{A4}$$

$$= \mathbb{E}_{q(\theta|\mathcal{D})}\left[\log \frac{q(\theta|\phi)}{p(\mathcal{D}|\theta)p(\theta)} + \log p(\mathcal{D})\right] \tag{A5}$$

$$= \mathbb{E}_{q(\theta|\mathcal{D})}\left[\log \frac{q(\theta|\phi)}{p(\theta)} - \log p(\mathcal{D}|\theta) + \log p(\mathcal{D})\right] \tag{A6}$$

$$= \mathbb{KL}[q(\theta|\phi)||p(\theta)] - \mathbb{E}_{q(\theta|\mathcal{D})}[\log p(\mathcal{D}|\theta)] + \log p(\mathcal{D}). \tag{A7}$$

Since the Kullback–Leibler divergence is greater or equal to zero at all times, we have

$$0 \geq \mathbb{KL}[q(\theta|\phi)||p(\theta)] - \mathbb{E}_{q(\theta|\mathcal{D})}[\log p(\mathcal{D}|\theta)] + \log p(\mathcal{D}) \tag{A8}$$

$$-\log p(\mathcal{D}) \leq \mathbb{KL}[q(\theta|\phi)||p(\theta)] - \mathbb{E}_{q(\theta|\mathcal{D})}[\log p(\mathcal{D}|\theta)] \tag{A9}$$

$$\log p(\mathcal{D}) \geq \mathbb{E}_{q(\theta|\mathcal{D})}[\log p(\mathcal{D}|\theta)] - \mathbb{KL}[q(\theta|\phi)||p(\theta)]. \tag{A10}$$

Appendix B. Ito's Lemma

Let X_t be an Ito drift-diffusion process that satisfies the SDE

$$dX_t = \mu_t dt + \sigma_t dW_t \tag{A11}$$

where W_t is a Wiener process. For a scalar function $f(X_t, t)$, which is twice differentiable in X_t and once differentiable in time t, we can apply the Taylor expansion up to the second-order to obtain

$$df(X_t, t) = \partial_t f(X_t, t) + \partial_X f(X_t, t) dX_t + \frac{1}{2}\partial_X^2 f(X_t, t) dX_t^2. \tag{A12}$$

We can then substitute the Ito drift-diffusion process dX_t into the Taylor expansion. In the limit of $dt \to 0$, the terms dt^2 and $dtdW_t$ tend to zero faster than $dW_t^2 = \sqrt{dt}^2$ due to their higher exponent. Multiplying out the terms and setting the relevant infinitessimal terms to zero, we obtain

$$df(X_t, t) = \partial_t f(X_t, t) dt + \partial_X f(X_t, t) dX_t + \frac{1}{2}\partial_X^2 f(X_t, t) dX_t^2 \tag{A13}$$

$$= \partial_t f(X_t, t) dt + \partial_X f(X_t, t)(\mu_t dt + \sigma_t dW_t) + \frac{1}{2}\partial_X^2 f(X_t, t)(\mu_t dt + \sigma_t dW_t)^2 \tag{A14}$$

$$= \partial_t f(X_t, t) dt + \partial_X f(X_t, t)(\mu_t dt + \sigma_t dW_t) \tag{A15}$$

$$+ \frac{1}{2}\partial_X^2 f(X_t, t)\big(\underbrace{\mu_t dt^2 + 2\mu_t \sigma_t dt dW_t}_{=0} + \sigma_t^2 \underbrace{dW_t^2}_{=dt}\big) \tag{A16}$$

$$= \partial_t f(X_t, t) dt + \partial_X f(X_t, t)(\mu_t dt + \sigma_t dW_t) + \frac{1}{2}\partial_X^2 f(X_t, t)\sigma_t^2 dt \tag{A17}$$

$$= \big(\partial_t f(X_t, t) + \mu_t \partial_X f(X_t, t) + \frac{1}{2}\partial_X^2 f(X_t, t)\sigma_t^2\big) dt + \sigma_t \partial_X f(X_t, t) dW_t \tag{A18}$$

which is again an Ito drift-diffusion process, albeit with more complex drift and diffusion terms.

Appendix C. Bayesian Stochastic Differential Equation of a Variational Distribution

To validate the derivation of the SDE that governs the dynamics of the variational parameters, we evaluate the proposed SDE on a simplified example before moving on to more general Bayesian models, such as deep Bayesian neural networks.

In this intuitive example, we model a parameter θ as a distribution $\theta \sim \mathcal{N}(\mu_t, \sigma_t(\rho_t))$ and reparameterize the uncertainty parameter $\sigma_t \in \mathbb{R}^+$ with an unbounded surrogate parameter $\rho_t \in \mathbb{R}$ [27,31–33]. The subscript t indicates the parameters at time t during the optimization process. By making use of the reparameterization trick, we obtain a sampling scheme which is differentiable with respect to the parameters $\phi_t = \{\mu_t, \rho_t\}$. For a normal distribution this takes the form

$$\theta_t = \mu_t + \epsilon\sigma_t(\rho_t) = \mu_t + \epsilon \log(1 + \exp(\rho_t)) \tag{A19}$$

with $\epsilon \sim \mathcal{N}(0, 1)$ being the externalized stochasticity.

The reparameterization allows us to compute the derivatives for any differentiable loss function $\mathcal{L}$ with respect to the variational parameters ϕ_t. In our simplified case, we proceed with the quadratic loss function

$$\mathcal{L} = \frac{1}{2}\theta_t^2 = \frac{1}{2}\big(\mu_t + \epsilon \log(1 + \exp(\rho_t))\big)^2 \tag{A20}$$

for which we can compute the gradients

$$\partial_\mu \mathcal{L} = \mu_t + \epsilon\ \sigma_t(\rho_t) \tag{A21}$$

$$\partial_\rho \mathcal{L} = \mu_t\ \epsilon\ \sigma(\rho_t) + \epsilon^2\ \sigma_t(\rho_t)\ \sigma(\rho_t) \tag{A22}$$

where $\sigma(\cdot)$ is the sigmoid function and $\sigma_t(\cdot)$ is the reparameterization of the uncertainty parameter ρ_t.

The first- and second-order moments of the gradients are

$$\mathbb{E}\big[\partial_\mu \mathcal{L}\big] = \mu_t \tag{A23}$$

$$\mathbb{E}\big[\partial_\rho \mathcal{L}\big] = \sigma_t(\rho_t)\ \sigma(\rho_t) \tag{A24}$$

and, with the fact that, by definition, $\mathbb{E}\big[\epsilon^2\big] = 1$,

$$\mathbb{V}\big[\mathbb{E}\big[\partial_\mu \mathcal{L}\big]\big] = \sigma_t(\rho_t)^2 \tag{A25}$$

$$\mathbb{V}\big[\mathbb{E}\big[\partial_\rho \mathcal{L}\big]\big] = \mathbb{V}[\mu_t\ \epsilon\ \sigma(\rho_t)] + \mathbb{V}\Big[\epsilon^2\ \sigma_t(\rho_t)\ \sigma(\rho_t)\Big] \tag{A26}$$

$$= \mu_t^2 \sigma(\rho)^2 + \mathbb{V}\Big[\epsilon^2\Big]\ \sigma_t(\rho_t)^2\ \sigma(\rho_t)^2 \tag{A27}$$

As the loss function $\mathcal{L}$ does not include any data uncertainty, the aleatoric uncertainty is reduced to zero. We can now expand the loss function $\mathcal{L}$ to include aleatoric uncertainty from the data. For this, we sample from two functions $\mathcal{L}_m$ which are the original cost function $\mathcal{L}$ shifted by $\pm b$, the purpose of which is to simulate data uncertainty that occurs in gradient optimization with mini-batches:

$$\mathcal{L}_1 = \frac{1}{2}(\theta_t - b)^2 - 1 \tag{A28}$$

$$= \frac{1}{2}(\mu_t + \epsilon \log(1 + e^{\rho_t}))^2 - b)^2 - 1 \tag{A29}$$

$$\mathcal{L}_2 = \frac{1}{2}(\theta_t + b)^2 - 1 \tag{A30}$$

$$= \frac{1}{2}(\mu_t + \epsilon \log(1 + e^{\rho_t}))^2 + b)^2 - 1. \tag{A31}$$

Furthermore, we shift each loss function $\mathcal{L}_m$ by -1, such that the local optima of the loss function are lower than the optimum that balances both loss functions. By computing

the gradients from randomly sampled $\mathcal{L}_1$ and $\mathcal{L}_2$ during training, we include a simplified version of mini-batch sampling from a dataset with two samples. Both $\mathcal{L}_1$ and $\mathcal{L}_2$ provide local minima at $\pm b$, while the global optimum still lies in the middle between them at 0.

We can now estimate the aleatoric variance $\mathbb{E}\big[\mathbb{V}\big[\nabla_\phi \mathcal{L}_m | \mathcal{L}_m\big]\big]$ in the gradient with

$$\mathbb{E}\big[\mathbb{V}\big[\partial_\mu \mathcal{L}_m | \mathcal{L}_m\big]\big] = \frac{1}{2}\sum_{i=1}^{2}\big(\partial_\mu \mathcal{L}_m(\mu, \rho_t) - \partial_\mu \mathcal{L}(\mu, \rho)\big)^2 \tag{A32}$$

$$= b^2 \tag{A33}$$

$$\mathbb{E}\big[\mathbb{V}\big[\partial_\rho \mathcal{L}_m | \mathcal{L}_m\big]\big] = \frac{1}{2}\sum_{i=1}^{2}\big(\partial_\rho \mathcal{L}_m(\mu, \rho) - \partial_\rho \mathcal{L}(\mu, \rho)\big)^2 \tag{A34}$$

$$= b^2\epsilon^2\sigma(\rho_t)^2 \tag{A35}$$

We can, thus, define the Ito drift-diffusion process for the variational parameters with the decomposed diffusion as

$$d\mu_t = -\mathbb{E}\big[\partial_\mu \mathcal{L}_m\big]dt + \big(\mathbb{V}\big[\mathbb{E}\big[\partial_\mu \mathcal{L}_m | \mathcal{L}_m\big]\big] + \mathbb{E}\big[\mathbb{V}\big[\partial_\mu \mathcal{L}_m | \mathcal{L}_m\big]\big]\big)^{\frac{1}{2}} dW_t \tag{A36}$$

$$d\rho_t = -\mathbb{E}\big[\partial_\rho \mathcal{L}_m\big]dt + \big(\mathbb{V}\big[\mathbb{E}\big[\partial_\rho \mathcal{L}_m | \mathcal{L}_m\big]\big] + \mathbb{E}\big[\mathbb{V}\big[\partial_\rho \mathcal{L}_m | \mathcal{L}_m\big]\big]\big)^{\frac{1}{2}} dW_t \tag{A37}$$

For this simplifying example, we choose to derive a decoupled set of SDEs. In both cases the chain rule passes the gradients through the sampled parameter θ_t. As it turns out, and as we make use of them in the subsequently derived stochastic optimal control optimziation algorithm, both $\partial_\mu \mathcal{L}$ and $\partial_\rho \mathcal{L}$ share the gradient $\partial_\theta \mathcal{L}$ when applying the chain rule,

$$\partial_\mu \mathcal{L} = \partial_\theta \mathcal{L} \partial_\mu \theta_t \tag{A38}$$

$$\partial_\rho \mathcal{L} = \partial_\theta \mathcal{L} \partial_\sigma \theta_t \partial_\rho \sigma \tag{A39}$$

Appendix D. Gradient Derivations

We employ the Einstein summation to derive the relevant gradients. We use the lower index for the horizontal indices and the upper index for the vertical indices of a matrix.

$$f(z) = z^\top A z = z_i A^i_j z^j \tag{A40}$$

$$\nabla f(z) = Az \quad = A^i_j z^j \tag{A41}$$

$$M = zz^T \quad = z^i z_j \tag{A42}$$

$$\nabla M = \left[\frac{\partial M^i_j}{\partial z^l}\right] = z_j \delta_{il} + z^i \delta_{jl} \tag{A43}$$

$$\nabla M \; U \; \nabla f = \Big(z_j \delta_{il} + z^i \delta_{jl}\Big) U^l_m A^m_n z^n \tag{A44}$$

$$= z_j \delta_{il} U^l_m A^m_n z^n + z^i \delta_{jl} U^l_m A^m_n z^n \tag{A45}$$

$$= z_j U^i_m A^m_n z^n + z^i U^j_m A^m_n z^n \tag{A46}$$

$$= U^i_m A^m_n z^n z_j + z^i z_n A^n_m U^m_i \tag{A47}$$

$$= UAM + MAU \tag{A48}$$

$$\frac{\partial^2 M_{ij}}{\partial z_l \partial z_m} = \delta_{il}\delta_{mj} + \delta_{im}\delta_{jl} \tag{A49}$$

$$\mathrm{Tr}\Big[DU\nabla^2 MU\Big] = \mathrm{Tr}\Big[D_p^q U_m^p \Big(\delta_{il}\delta_{mj} + \delta_{im}\delta_{jl}\Big) U_k^l\Big] \tag{A50}$$
$$= \mathrm{Tr}\Big[D_p^q U_m^p \delta_{mj}\delta_{il} U_{lk} + D_p^q U_m^p \delta_{im}\delta_{jl} U_k^l\Big] \tag{A51}$$
$$= \mathrm{Tr}\Big[D_p^q U_j^p U_k^i + D_p^q U_i^p U_k^j\Big] \tag{A52}$$
$$= \mathrm{Tr}\Big[D_p^q U_j^p U_k^i\Big] + \mathrm{Tr}\Big[D_p^q U_i^p U_k^j\Big] \tag{A53}$$
$$= D_p^q U_j^p U_q^i + D_p^q U_i^p U_q^j \tag{A54}$$
$$= U_q^i D_p^q U_j^p + U_q^j D_p^q U_i^p \tag{A55}$$
$$= UDU + (UDU)^T \tag{A56}$$
$$= 2UDU \tag{A57}$$

Appendix E. Stochastic Control

We have the dynamics given as

$$\dot{M} = -(UAM + MAU) + \eta UDU \tag{A58}$$

We want to minimize the final cost

$$C(M, U, t_1) = \frac{1}{2}\mathrm{Tr}[AM]. \tag{A59}$$

Alternatively, we can directly minimize the dynamics $\dot{M}$ over time, such that

$$\min_U C = \min_U \frac{1}{2}\int_{t_0}^{t_1} \mathrm{Tr}\big[A\dot{M}\big]dt \tag{A60}$$

which requires us to minimize $\mathrm{Tr}\big[A\dot{M}\big]$ at every step. Both A and M should be positive semi-definite to guarantee a bound from below.

Optimizing $\frac{1}{2}\mathrm{Tr}\big[A\dot{M}\big]$ gives us

$$\frac{1}{2}\mathrm{Tr}\big[A\dot{M}\big] = -\frac{1}{2}\mathrm{Tr}[AUAM + AMAU] + \frac{\eta}{2}\mathrm{Tr}[AUDU] \tag{A61}$$
$$= -\frac{1}{2}(\mathrm{Tr}[AUAM] + \mathrm{Tr}[AMAU]) + \frac{\eta}{2}\mathrm{Tr}[AUDU] \tag{A62}$$
$$= -\frac{1}{2}(\mathrm{Tr}[AMAU] + \mathrm{Tr}[AMAU]) + \frac{\eta}{2}\mathrm{Tr}[AUDU] \tag{A63}$$
$$= -\mathrm{Tr}[AMAU] + \frac{\eta}{2}\mathrm{Tr}[AUDU] \tag{A64}$$

The control matrix U is diagonal. The trace is defined as $\mathrm{Tr}[AB] = A_j^i B_i^j$ and the noise matrix is symmetric by definition, so $D_i^j = D_j^i$. In index notation, we can arbitrarily shuffle the individual terms in a product to obtain generalizable formulations in terms of matrices and vectors. This gives us the index notation of

$$-\mathrm{Tr}[AMAU] + \frac{\eta}{2}\mathrm{Tr}[AUDU] = -\sum_i (AMA)_i^i\, U_i^i + \frac{\eta}{2}\sum_{i,j} A_j^i U_j^j D_i^j U_i^i \tag{A65}$$
$$= -\sum_i (AMA)_i^i\, U_i^i + \frac{\eta}{2}\sum_{i,j} U_i^i \underbrace{A_j^i D_j^i}_{Q_j^i} U_j^j \tag{A66}$$
$$= -\sum_i (AMA)_i^i\, U_i^i + \frac{\eta}{2}\sum_{i,j} U_i^i Q_j^i U_j^j \tag{A67}$$

Taking the derivative with respect to the control parameters U_i^i, and setting it to zero, yields

$$0 = -(AMA)_i^i + \eta \sum_j Q_j^i U_j^j \tag{A68}$$

$$0 = -\text{Diag}[AMA] + \eta\, Q\, U' \tag{A69}$$

$$U' = Q^{-1} \frac{\text{Diag}[AMA]}{\eta} \tag{A70}$$

$$= (A \circ D)^{-1} \frac{\text{Diag}[AMA]}{\eta} \tag{A71}$$

where $\text{Diag}[A]$ is a vector of the diagonal elements of the matrix A and $U' = [U_1, U_2]^\top$ is a vector of the individual control parameters U_1 and U_2.

Appendix F. Estimation of Local Quadratic Approximation

We compute the offset b and the curvature A via the following relations:

$$\partial_b \left[\frac{1}{2}\mathbb{E}\Big[(\partial_\phi \mathcal{L} - A(x+b))^2\Big]\right] = \partial_b \left[\frac{1}{2}\mathbb{E}\Big[(\partial_\phi \mathcal{L} - Ax + Ab))^2\Big]\right] \tag{A72}$$

$$= \mathbb{E}\Big[A^T(\partial_\phi \mathcal{L} - Ax + Ab)\Big] \tag{A73}$$

$$= A^T \mathbb{E}\big[\partial_\phi \mathcal{L}\big] - A^T A \mathbb{E}[x] + A^T Ab \stackrel{!}{=} 0 \tag{A74}$$

$$\Downarrow \tag{A75}$$

$$A^T Ab = A^T A \mathbb{E}[x] - A^T \mathbb{E}\big[\partial_\phi \mathcal{L}\big] \tag{A76}$$

$$b = \mathbb{E}[x] - (A^T A)^{-1} A^T \mathbb{E}\big[\partial_\phi \mathcal{L}\big] \tag{A77}$$

$$b = \mathbb{E}[x] - A^{-1} A^{T^{-1}} A^T \mathbb{E}\big[\partial_\phi \mathcal{L}\big] \tag{A78}$$

$$b = \mathbb{E}[x] - A^{-1} \mathbb{E}\big[\partial_\phi \mathcal{L}\big] \tag{A79}$$

$$\partial_A \left[\frac{1}{2}\mathbb{E}\Big[(\partial_\phi \mathcal{L} - A(x+b))^2\Big]\right] = -\mathbb{E}\Big[\partial_\phi \mathcal{L}(x+b)^T - A(x+b)(x+b)^T\Big] \stackrel{!}{=} 0. \tag{A80}$$

Writing out the products, taking the expectations where necessary, and following through with the long and tedious algebra, we finally arrive at

$$\partial_A \left[\frac{1}{2} \mathbb{E}\Big[(\partial_\phi \mathcal{L} - A(x-b))^2 \Big] \right] \tag{A81}$$

$$= -\mathbb{E}\Big[\big(\partial_\phi \mathcal{L} - A(x-b)\big) \big(x-b\big)^T \Big] \tag{A82}$$

$$= -\mathbb{E}\Big[\partial_\phi \mathcal{L}(x-b)^T - A(x-b)(x-b)^T \Big] \tag{A83}$$

$$= -\mathbb{E}\Big[\partial_\phi \mathcal{L} x^T - \partial_\phi \mathcal{L}(\mathbb{E}[x] - A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big])^T - A(xx^T - bx^T - xb^T + bb^T) \Big] \tag{A84}$$

$$\begin{aligned} = -\mathbb{E}\Big[& \partial_\phi \mathcal{L} x^T - \partial_\phi \mathcal{L}\mathbb{E}[x]^T + \partial_\phi \mathcal{L}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1} \\ & - A\Big(xx^T - \Big(\mathbb{E}[x] - A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\Big)x^T - x\Big(\mathbb{E}[x] - A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\Big)^T \\ & + \Big(\mathbb{E}[x] - A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\Big)\Big(\mathbb{E}[x] - A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\Big)^T \Big) \Big] \end{aligned} \tag{A85}$$

$$\begin{aligned} = -\Big[& \mathbb{E}\Big[\partial_\phi \mathcal{L} x^T\Big] - \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T + \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1} \\ & - A\Big(\mathbb{E}\Big[xx^T\Big] - \mathbb{E}[x]\mathbb{E}[x]^T + \cancel{A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T} - \cancel{\mathbb{E}[x]\mathbb{E}[x]^T} \\ & + \cancel{\mathbb{E}[x]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1}} + \cancel{\mathbb{E}[x]\mathbb{E}[x]^T} - \cancel{\mathbb{E}[x]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1}} \\ & - \cancel{A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T} + A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1} \Big) \Big] \end{aligned} \tag{A86}$$

$$\begin{aligned} = -\Big[& \mathbb{E}\Big[\partial_\phi \mathcal{L} x^T\Big] - \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T + \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1} \\ & - A\Big(\mathbb{E}\Big[xx^T\Big] - \mathbb{E}[x]\mathbb{E}[x]^T + A^{-1}\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1} \Big) \Big] \end{aligned} \tag{A87}$$

$$\begin{aligned} = -\Big[& \mathbb{E}\Big[\partial_\phi \mathcal{L} x^T\Big] - \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T + \cancel{\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1}} \\ & - A\Big(\mathbb{E}\Big[xx^T\Big] - \mathbb{E}[x]\mathbb{E}[x]^T \Big) - \cancel{\mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}\big[\partial_\phi \mathcal{L}\big]^T A^{-1}} \Big] \end{aligned} \tag{A88}$$

$$= -\Big(\mathbb{E}\Big[\partial_\phi \mathcal{L} x^T\Big] - \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T \Big) + A\Big(\mathbb{E}\Big[xx^T\Big] - \mathbb{E}[x]\mathbb{E}[x]^T \Big) \tag{A89}$$

$$\stackrel{!}{=} 0 \tag{A90}$$

$$\Downarrow \tag{A91}$$

$$A = \Big(\mathbb{E}\Big[\partial_\phi \mathcal{L} x^T\Big] - \mathbb{E}\big[\partial_\phi \mathcal{L}\big]\mathbb{E}[x]^T \Big) \Big(\mathbb{E}\Big[xx^T\Big] - \mathbb{E}[x]\mathbb{E}[x]^T \Big)^{-1} \tag{A92}$$

As we deal with matrices in $\mathbb{R}^{2\times 2}$ per set of variational parameters $\phi = \{\mu, \sigma\}$, the matrix inversions can be performed cheaply and analytically with the identity

$$A^{-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} \tag{A93}$$

which requires only two multiplications, one subtraction and a repositioning of values with two negations.

Appendix G. Experimental Setup

In all the experiments, we trained a convolutional Bayesian neural network (CBNN) with four blocks of 2D convolution, 2D BatchNorm and 2D MaxPooling, followed by two linear layers. The convolutional filters were in the sequence 96, 128, 256 and 128 and the subsequent layers had 200 and 10 neurons, respectively. This corresponds to the

experimental setup of [15]. As a baseline, we optimized a Bayesian feed-forward neural network (BNN) with four layers and 200 neurons in each layer. As a frequentist baseline, we trained the same architectures but without any regularization compared to the KL divergence in the ELBO for Bayesian neural networks. We used a consistent batch size of 64 due to the memory constraints of computing per sample gradients and used leaky ReLU with a negative slope of 0.01 in all architectures. The convolutional and linear layer parameters were parameterized with normal distributions and the ELBO was used as the objective function.

Experimentally and theoretically, it has been found that large learning rates in the early phase of optimization help to provide a large approximate improvement before decreasing the learning rate to finetune the model parameters [34,35]. For our experiments, we initialized the control parameter with a value of one, which corresponds to adaptive learning rate scheduling [26]. Similarly to the learning rate, learning rate scheduling requires an a priori decision on how the scheduling should be conducted. Our optimizer StochControlSGD adaptively determines the optimal learning rate via the principle of stochastic optimal control independently for each parameter .

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; Volume 25, pp. 1097–1105.
2. Hannun, A.; Case, C.; Casper, J.; Catanzaro, B.; Diamos, G.; Elsen, E.; Prenger, R.; Satheesh, S.; Sengupta, S.; Coates, A.; et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv* **2014**, arXiv:1412.5567.
3. Brown, T.B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *arXiv* **2020**, arXiv:2005.14165.
4. Andrieu, C.; De Freitas, N.; Doucet, A.; Jordan, M.I. An introduction to MCMC for machine learning. *Mach. Learn.* **2003**, *50*, 5–43.
5. Wainwright, M.J.; Jordan, M.I. *Graphical Models, Exponential Families, and Variational Inference*; Now Publishers Inc.: Norwell, MA, USA, 2008.
6. Hoffman, M.D.; Blei, D.M.; Wang, C.; Paisley, J. Stochastic variational inference. *J. Mach. Learn. Res.* **2013**, *14*, 1303–1347.
7. Bottou, L. Large-scale machine learning with stochastic gradient descent. In Proceedings of the COMPSTAT'2010: 19th International Conference on Computational Statistics, Paris, France, 22–27 August 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 177–186.
8. Liu, G.H.; Theodorou, E.A. Deep learning theory review: An optimal control and dynamical systems perspective. *arXiv* **2019**, arXiv:1908.10920.
9. Orvieto, A.; Kohler, J.; Lucchi, A. The role of memory in stochastic optimization. In Proceedings of the Uncertainty in Artificial Intelligence (PMLR), Virtual, 3–6 August 2020; pp. 356–366.
10. Mandt, S.; Hoffman, M.D.; Blei, D.M. Stochastic gradient descent as approximate Bayesian inference. *arXiv* **2017**, arXiv:1704.04289.
11. Yaida, S. Fluctuation-dissipation relations for stochastic gradient descent. *arXiv* **2018**, arXiv:1810.00004.
12. Oksendal, B. *Stochastic Differential Equations: An Introduction with Applications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2013.
13. Depeweg, S.; Hernandez-Lobato, J.M.; Doshi-Velez, F.; Udluft, S. Decomposition of uncertainty in Bayesian deep learning for efficient and risk-sensitive learning. In Proceedings of the International Conference on Machine Learning (PMLR), Stockholm, Sweden, 10–15 July 2018; pp. 1184–1193.
14. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3th International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
15. Li, Q.; Tai, C.; Weinan, E. Stochastic modified equations and adaptive stochastic gradient algorithms. In Proceedings of the International Conference on Machine Learning (PMLR), Sydney, Australia, 6–11 August 2017; pp. 2101–2110.
16. Stengel, R.F. *Optimal Control and Estimation*; Courier Corporation: Chelmsford, MA, USA, 1994.
17. LeCun, Y. The MNIST Database of Handwritten Digits. 1998. Available online: http://yann.lecun.com/exdb/mnist/ (accessed on 04 March 2022)).
18. Xiao, H.; Rasul, K.; Vollgraf, R. Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. *arXiv* **2017**, arXiv:1708.07747.
19. Krizhevsky , A.; Hinton, G. Convolutional Deep Belief Networks on Cifar-10. 2010. Available online: https://www.cs.toronto.edu/~kriz/conv-cifar10-aug2010.pdf (accessed on 04 March 2022)).
20. Wenzel, F.; Roth, K.; Veeling, B.S.; Swiatkowski, J.; Tran, L.; Mandt, S.; Snoek, J.; Salimans, T.; Jenatton, R.; Nowozin, S. How good is the bayes posterior in deep neural networks really? *arXiv* **2020**, arXiv:2002.02405.
21. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the International Conference on Machine Learning (PMLR), Lille, France, 7–9 July 2015; pp. 448–456.
22. Frankle, J.; Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv* **2018**, arXiv:1803.03635.

23. Smith, S.L.; Kindermans, P.J.; Ying, C.; Le, Q.V. Don't decay the learning rate, increase the batch size. *arXiv* **2017**, arXiv:1711.00489.
24. Murata, N.; Kawanabe, M.; Ziehe, A.; Müller, K.R.; Amari, S.i. On-line learning in changing environments with applications in supervised and unsupervised learning. *Neural Netw.* **2002**, *15*, 743–760.
25. Murata, N.; Müller, K.R.; Ziehe, A.; Amari, S.-i. Adaptive on-line learning in changing environments. In Proceedings of the Advances in Neural Information Processing Systems, Denver, CO, USA, 01–06 December 1997; pp. 599–605.
26. Loshchilov, I.; Hutter, F. Sgdr: Stochastic gradient descent with warm restarts. *arXiv* **2016**, arXiv:1608.03983.
27. Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; Wierstra, D. Weight uncertainty in neural network. In Proceedings of the International Conference on Machine Learning (PMLR), Lille, France, 7–9 July 2015; pp. 1613–1622.
28. Gal, Y.; Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Proceedings of the International Conference on Machine Learning (PMLR), New York, NY, USA, 20–22 June 2016; pp. 1050–1059.
29. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
30. Kingma, D.P.; Salimans, T.; Welling, M. Variational dropout and the local reparameterization trick. *arXiv* **2015**, arXiv:1506.02557.
31. Ranganath, R.; Gerrish, S.; Blei, D. Black box variational inference. In Proceedings of the Artificial Intelligence and Statistics (PMLR), Bejing, China, 22–24 June 2014; pp. 814–822.
32. Baydin, A.G.; Pearlmutter, B.A.; Radul, A.A.; Siskind, J.M. Automatic differentiation in machine learning: A survey. *J. Mach. Learn. Res.* **2018**, *18*, 1–43.
33. Kucukelbir, A.; Tran, D.; Ranganath, R.; Gelman, A.; Blei, D.M. Automatic differentiation variational inference. *J. Mach. Learn. Res.* **2017**, *18*, 430–474.
34. Robbins, H.; Monro, S. A stochastic approximation method. *Ann. Math. Stat.* **1951**, *22*, 400–407.
35. Welling, M.; Teh, Y.W. Bayesian learning via stochastic gradient Langevin dynamics. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 681–688.