IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 4 May 2023, accepted 16 May 2023, date of publication 23 May 2023, date of current version 7 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3279124

RESEARCH ARTICLE

On Performance and Calibration of Natural Gradient Langevin Dynamics

HANIF AMAL ROBBANI[1], ALHADI BUSTAMAM[1], RISMAN ADNAN[1,2], AND SHANDAR AHMAD[3]

[1]Department of Mathematics, Universitas Indonesia, Depok 16424, Indonesia
[2]Kalbe Digital Lab, Jakarta 10510, Indonesia
[3]School of Computational and Integrative Sciences, Jawaharlal Nehru University, Delhi 110067, India

Corresponding author: Alhadi Bustamam (alhadi@sci.ui.ac.id)

This work was supported by the International Indexed Publication (PUTI) Q1 2022 Research Grant from Directorate of Research and Development Universitas Indonesia under Contract NKB-471/UN2.RST/HKP.05.00/2022.

ABSTRACT Producing deep neural network (DNN) models with calibrated confidence is essential for applications in many fields, such as medical image analysis, natural language processing, and robotics. Modern neural networks have been reported to be poorly calibrated compared with those from a decade ago. The stochastic gradient Langevin dynamics (SGLD) algorithm offers a tractable approximate Bayesian inference applicable to DNN, providing a principled method for learning the uncertainty. A recent benchmark study showed that SGLD could produce a more robust model to covariate shifts than other competing methods. However, vanilla SGLD is also known to be slow, and preconditioning can improve SGLD efficacy. This paper proposes eigenvalue-corrected Kronecker factorization (EKFAC) preconditioned SGLD (EKSGLD), in which a novel second-order gradient approximation is employed as a preconditioner for the SGLD algorithm. This approach is expected to bring together the advantages of both second-order optimization and the approximate Bayesian method. Experiments were conducted to compare the performance of EKSGLD with existing preconditioning methods and showed that it could achieve higher predictive accuracy and better calibration on the validation set. EKSGLD improved the best accuracy by 3.06% on CIFAR-10 and 4.15% on MNIST, improved the best negative log-likelihood by 16.2% on CIFAR-10 and 11.4% on MNIST, and improved the best thresholded adaptive calibration error by 4.05% on CIFAR-10.

INDEX TERMS Natural gradient, second-order optimization, Bayesian deep learning, Langevin dynamics, confidence calibration, predictive uncertainty.

I. INTRODUCTION

The advances in deep learning have been remarkable, showing the ability to achieve high-performance accuracy in a wide range of areas, such as natural language processing [7], computer vision, and medical diagnosis [19]. Consequently, DNNs are now entrusted to take an important part in complex decision-making pipelines in those fields.

Although many successes have been reported so far, effectively training the DNNs is still challenging because the objective function to be optimized has a pathological curvature and a highly nonconvex nature. Furthermore, the loss surface is known to have highly imbalanced curvature [10]. These limit the efficiency of commonly used first-order gradient-based optimization algorithms such as stochastic gradient descent (SGD). Methods that apply second-order information have the potential to accelerate first-order gradient descent by correcting the imbalanced curvature. The process involves a preconditioning matrix that captures the local curvature or related information, such as the Hessian matrix in Newton's method or the Fisher information matrix (FIM) in natural gradient [3]. Unfortunately, the size of the preconditioning matrix becomes gigantic in most DNN setups, making these methods impractical using their original form.

The associate editor coordinating the review of this manuscript and approving it for publication was Qiang Li.

Various algorithms used for optimizing DNNs can be interpreted as approximating the diagonal of a large preconditioning matrix. Despite being efficient, these algorithms are considered crude approximations since they ignore correlations between parameters. A refined algorithm must consider some correlations between different parameters. Kronecker factored approximate curvature (K-FAC) uses block-diagonal approximation, where each block corresponds to a layer in the DNNs [28]. K-FAC is derived by approximating each large block as the Kronecker product of two much smaller matrices. Approximating and inverting the two smaller matrices are much more efficient than doing so on the whole block matrix. A further improved version of K-FAC is eigenvalue-corrected Kronecker factorization (EKFAC), which tracks diagonal variance in a Kronecker-factored eigenbasis instead of in the parameter coordinates. EKFAC provides a better approximation of the FIM than the K-FAC, which may produce parameter updates closer to the exact natural gradient [14].

In real-world applications with high stakes, such as automated medical diagnosis and self-driving cars, calibrated confidence is especially important besides prediction accuracy. For example, in automated medical diagnosis, human doctors should make decisions when the confidence in a disease diagnosis by DNNs is low. However, modern DNNs with significantly deeper and wider layers tend to yield overconfident predictions [12], [17]. One popular approach to address this issue is the recalibration of probabilities on a held-out validation set using histogram binning [45], temperature scaling [17], [37], isotonic regression [46], and other similar methods. As an alternative, Bayesian methods for DNNs provide a natural mechanism to represent uncertainty, potentially leading to improved generalization and calibrated predictive distributions [21], [35]. However, typical Bayesian methods using classical Markov chain Monte Carlo (MCMC) algorithms such as full-batch Hamiltonian Monte Carlo (HMC) [31] require expensive computations over the entire dataset, making their application to DNNs difficult. Therefore, further research on practical Bayesian methods is necessary.

SGLD offers a tractable approximate Bayesian inference applicable to DNNs, which theoretically provides in-built protection against overfitting [43]. A recent benchmark shows that SGLD produces a predictive distribution as close to the gold-standard HMC as the more popular method, deep ensemble. SGLD, along with the deep ensemble, has also been shown to be more robust to covariate shifts than HMC [21]. These results suggest that SGLD is a strong candidate for addressing these challenges. Despite the preferable properties mentioned, vanilla SGLD is also known to be slow. Training with vanilla SGLD is normally done with a very small learning rate over a large number of iterations. Incorporating a preconditioning matrix similar to RMSProp was proposed to improve the efficacy of SGLD in a method named preconditioned SGLD (pSGLD) [27]. A recent proposal uses K-FAC as a preconditioner for SGLD and shows that this technique produces better sampling when compared to pSGLD [29]. These prior works on SGLD with preconditioning, or what we can interpret as approximate natural gradient Langevin Dynamics (NGLD), also lack performance evaluation on uncertainty calibration and robustness to dataset shift.

This work proposes an improved implementation of SGLD using EKFAC preconditioning. This technique is expected to bring advantages from second-order optimization and approximate Bayesian methods. This novel method is referred to as EKFAC preconditioned SGLD (EKSGLD). This work empirically demonstrates that EKSGLD gives better model accuracy when compared with other SGLD preconditioning methods after the same number of iterations.

The scope and experimental design in this work were inspired by the previous works that 1) benchmarked various approximate NGLD methods [36], and 2) evaluated the predictive uncertainty produced by different probabilistic deep learning methods, as well as their robustness to dataset shift [35]. Besides proposing a new preconditioning approach, this work presents an empirical evaluation of predictive confidence calibration of approximate NGLD methods and its robustness to dataset shift. The results fill in the gap left by the original papers of the approximate NGLD methods, which do not evaluate uncertainty calibration and robustness to dataset shift, and the Bayesian deep learning benchmark papers that rarely include NGLD methods. Experiments show that EKSGLD produces better-calibrated confidence compared with its closest predecessor, KSGLD, on i.i.d. and out-of-distribution (OOD) test datasets. The terms confidence calibration and predictive uncertainty quality are used interchangeably throughout this paper.

A. CONTRIBUTION

Our study yields the following contributions:

- We demonstrate the effectiveness of EKFAC as a preconditioner for the SGLD optimization algorithm. We show that EKSGLD produces a model with better classification performance than the existing SGLD preconditioning methods after training with either the same number of epochs or the same time duration.
- We provide a performance comparison of approximate NGLD methods on two different image datasets: MNIST and CIFAR-10.
- We report the confidence calibration quality of models trained using approximate NGLD methods on three different types of test datasets, namely, i.i.d., shifted, and OOD test datasets.

B. RELATED WORK

In recent years, Bayesian deep learning (BDL) is gaining more attention due to its promising potential to estimate uncertainty based on solid theoretical principles. Many papers propose the application of BDL in a wide range of fields, from medical image classification in healthcare to data analysis from wearable devices and automatic assembly lines

in manufacturing. Other papers have been published doing a both theoretical and empirical examination of existing BDL methods or proposing improvement or new practical BDL methods [11], [18], [21], [23], [34], [44]. SGLD is one of the methods that are often included in BDL evaluation or benchmark experiments.

Since the introduction of SGLD [43], several modifications have been proposed to improve the efficacy of vanilla SGLD. One of the methods is based on a preconditioning matrix that approximates FIM. Previously, in the area of second-order optimization, the natural gradient method already used the FIM preconditioner [3]. The first paper that proposes a preconditioning method for SGLD uses a diagonal approximation of inverse FIM based on the RMSProp algorithm [27]. To give a better approximation of the inverse FIM while keeping the computation and storage consumption efficient, the K-FAC was proposed, which uses block-diagonal approximation [16], [28]. K-FAC was then adopted as a preconditioning matrix in SGLD and demonstrated its effectiveness for regression tasks in a small-scale experiment [29]. Recently, another adaptive preconditioner was proposed based on a diagonal approximation of second-order moment of gradient updates. This method is called adaptively preconditioned stochastic gradient Langevin dynamics (ASGLD) [6].

To the best of our knowledge, most, if not all, BDL evaluations published so far only include SGLD without preconditioning or did not include SGLD at all in favor of more popular BDL methods such as deep ensemble and variational inference (VI). Table 1 summarizes some prior works related to BDL benchmarks on image classification tasks. In this work, we focus on the empirical examination of SGLD and its variations that use different preconditioning methods to approximate the inverse of FIM. Below, we will elaborate more on these prior works related to the BDL benchmark, especially on image classification tasks, and also overview some of the most recent applications of BDL methods in various fields.

TABLE 1. Comparison with other BDL benchmarks on image classification tasks.

Ref.	Method	Metric	Dataset
Filos [11]	Deep ensemble Ensemble MC-Dropout MC-Dropout MFVI	Accuracy AUC	Kaggle EyePACS
Izmailov [21]	Deep ensemble HMC MFVI SGLD	Accuracy ECE NLL	CIFAR-10 CIFAR-100
Osawa [34]	BBB MC-Dropout Noisy K-FAC OGN VOGN	Accuracy AUC ECE NLL	CIFAR-10 ImageNet
Ovadia [35]	Deep ensemble LL-Dropout LL-SVI MC-Dropout SVI	Accuracy Brier ECE	CIFAR-10 ImageNet MNIST
Vadera [42]	Distilled SGHMC HMC MC-Dropout SGHMC SGLD Subspace Inference SWAG	Accuracy AUC AUPR Brier ECE NLL	CIFAR-10 CIFAR-100 ImageNet MNIST
Palacci [36]	KSGLD pSGLD SGLD	Accuracy	MNIST
This work	ASGLD EKSGLD KSGLD pSGLD SGLD	Accuracy AUC_{μ} ECE* NLL	CIFAR-10 MNIST

* Please refer to Section IV for the complete calibration metrics list.

1) BDL BENCHMARK ON IMAGE CLASSIFICATION TASK

Palacci and Hess compared the performance of vanilla SGLD, SGLD with RMSProp preconditioning, and SGLD with K-FAC preconditioning on MNIST classification and OOD sample detection tasks [36]. Izmailov et al. compared the performance of vanilla SGLD with HMC, mean-field VI (MFVI), and deep ensemble for training the ResNet model on CIFAR-10 and CIFAR-100 datasets. They evaluated the result on i.i.d. and shifted datasets and concluded that SGLD shows competitive performance in terms of accuracy and calibration compared with the other BDL methods on i.i.d. dataset, and SGLD along with deep ensemble is especially more robust compared to HMC on shifted dataset [21]. The most recent BDL benchmark we found is that of Vadera et al.. The paper presents a BDL benchmark framework to assess uncertainty, robustness, scalability, and accuracy named URSABench. The benchmark is done in three different scales: small (using MNIST dataset), medium (using CIFAR-10 and CIFAR-100 datasets), and large (using ImageNet dataset). Trained models are evaluated on i.i.d. and OOD test datasets. They concluded that SGLD and stochastic gradient Hamiltonian Monte Carlo (SGHMC) show the best performance overall [42].

The following works did not include SGLD in their BDL benchmark reports. Filos et al. compared BDL methods on a specific medical task of detecting diabetic retinopathy disease from fundus images which claimed to represent a real-world task better. They compared MC-Dropout, deep ensemble, MFVI, and ensemble MC-Dropout, and evaluated the result on i.i.d. and shifted datasets. They used a completely disjoint fundus image dataset collected with different medical equipment on a different population to represent a shifted dataset. They concluded that ensemble MC-Dropout performed consistently better on both i.i.d. and shifted test datasets [11].

Osawa et al. compared Bayes-by-backprop, MC-Dropout, and a natural gradient VI method called variational online Gauss-Newton (VOGN) for training models on CIFAR-10 and ImageNet datasets. Besides evaluating on i.i.d. set, they evaluated the result on the OOD dataset using SVHN and LSUN for models that were trained on CIFAR-10. They showed that VOGN performed best on 10 out of 15 metrics

on the i.i.d. dataset [34]. Ovadia et al. compared MC-Dropout, deep ensemble, stochastic VI (SVI), last layer (LL) SVI, and LL dropout for training models on MNIST, CIFAR-10, and ImageNet datasets. They evaluated the result on the shifted dataset and also on the OOD dataset using notMNIST and SVHN. They concluded that the accuracy and the quality of uncertainty consistently degrade with increasing dataset shift for all of the methods, and better calibration on the i.i.d. dataset is not usually followed by better calibration under dataset shift. Overall, the deep ensemble performed the best over most metrics and was more robust to dataset shift [35].

2) RECENT BDL APPLICATIONS

Healthcare is one of the areas where BDL methods have been applied in a wide range of tasks and data modalities. Gour et al. used MC-Dropout and EfficientNet neural architecture to build an uncertainty-aware model for the classification of coronavirus disease 2019 (COVID-19) based on chest X-ray images. The proposed method outperforms existing approaches in terms of classification performance. The model also provides calibrated uncertainty that is useful in the computer-aided diagnosis system for COVID-19 detection [15]. Song et al. used MC-Dropout in a VGG19-based model for oral cancer detection based on intraoral images. The accuracy of the model predictions increases by more than 4% when predictions with uncertainty greater than 0.3, or 10% of the predictions with the highest uncertainty scores, are discarded (to be referred to a human expert for further analysis) [40]. In the medical image segmentation tasks, Largent et al. used MC-Dropout and U-Net architecture as baseline models for automatic brain segmentation in preterm infants. The proposed method shows the best segmentation results across all tested methods and produces accurate uncertainty maps [25].

In the medical signal processing tasks, MC-Dropout and VI were used in the detection and classification of heart dysfunctions diseases based on electrocardiogram data [4], [20]. Previously, Fruehwirt et al. demonstrated that HMC outperforms MC-Dropout and non-Bayesian NN in Alzheimer's disease diagnosis based on electroencephalogram data [13]. In the electronic health record data analysis, Li et al. proposed a combination of the Gaussian process and VI to predict the first incidence of heart failure, diabetes, and depression. The result shows a better uncertainty modeling that is less susceptible to making overconfident predictions, even in the case of a minority class in imbalanced datasets. For a comprehensive review of the latest BDL applications in healthcare, we refer the reader to [1].

Landeghem et al. proposed a combination of deep ensemble and concrete dropout to model predictive uncertainty in natural language processing, specifically in multiclass and multilabel text classification tasks. The proposed method shows superior performance in calibration on i.i.d data, cross-domain classification, and novel class robustness [24]. Rodríguez-Puigvert et al. apply MC-Dropout in all layers of the DCNN-based encoder to produce better uncertainty quantification for robotic perception. The proposed method performed similarly well as the deep ensemble but with a smaller memory footprint [38].

Activities of daily living (ADLs) recognition systems play an important role in many applications, such as physical fitness monitoring, diet monitoring, and remote health monitoring. ADL recognition model trained on a certain user may not generalize well to new users due to variations in how people perform specific activities. Therefore, it is necessary to personalize underlying machine learning models to new users. Akbari and Jafari used MC-Dropout with variational autoencoder for personalizing ADL recognition systems with minimal solicitation of inputs or labels from users [2].

II. PRELIMINARIES

A. STOCHASTIC GRADIENT LANGEVIN DYNAMICS

Let θ denote a parameter vector, with $p(\theta)$ a prior distribution and $p(x|\theta)$ the probability of data item x given our model parameterized by θ. The posterior distribution of a set of N data items $X = \{x_i\}_{i=1}^N$ is: $p(\theta|X) \propto p(\theta)\Pi_{i=1}^N p(x_i|\theta)$. In the optimization literature, the prior regularizes the parameters, whereas the likelihood terms constitute the cost function to be optimized, and the task is to find the maximum a posteriori parameters θ^*. The SGD operates as follows. At each iteration t, a subset of n data items $X_t = \{x_{t1}, \ldots, x_{tn}\}$ is given, and the parameters are updated as follows:

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(x_{ti}|\theta_t)\right), \quad (1)$$

where ϵ_t is the step size at iteration t. The general idea is that the gradient calculated on the subset will be used to approximate the true gradient over the entire dataset.

SGLD combines the idea of SGD and Langevin dynamics by adding an amount of Gaussian noise balanced with the step size, allowing step sizes to go to 0:

$$\Delta\theta_t = \frac{\epsilon_t}{2}\left(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(x_{ti}|\theta_t)\right) + \eta_t$$
$$\eta_t \sim \mathcal{N}(0, \epsilon_t), \quad (2)$$

where the step sizes decrease toward 0. This enables averaging out of the stochasticity in the gradients and decreases Metropolis-Hastings (MH) rejection rates to zero asymptotically, so that one can simply ignore the MH acceptance steps, which require the calculation of probabilities over the entire dataset, altogether [43].

B. NATURAL GRADIENT LANGEVIN DYNAMICS

Given a dataset containing examples (x, y) and a DNN $f_\theta(x)$ with parameter vector θ of size n_θ, the SGD performs the first-order update rule: $\theta \leftarrow \theta - \eta\nabla_\theta$, where η is a positive learning rate. The second-order methods first modify the gradient ∇_θ by a preconditioning matrix G^{-1} resulting in the update rule of $\theta \leftarrow \theta - \eta G^{-1}\nabla_\theta$.

The space formed by the parameters of a probability distribution is a Riemannian manifold [3]. Its Riemannian metric is the FIM. This means that the parameter space is curved and that a local measure of curvature is the FIM. Natural gradient [3] uses FIM as preconditioning matrix G, which allows for adaptive gradient update and faster convergence in less number of iterations. Unfortunately, FIM has the size of $n_\theta \times n_\theta$, which, in many practical deep learning scenarios, is too big to compute and invert, hence requiring further approximation to make it more practical.

NGLD applies the same principles of using FIM, or approximation of FIM, as preconditioner in the SGLD settings:

$$\Delta\theta_t = \frac{\epsilon_t}{2} F^{-1}\left(\nabla \log p(\theta_t) + \frac{N}{n}\sum_{i=1}^{n} \nabla \log p(x_{ti}|\theta_t)\right) + F^{-1}\eta_t$$
$$\eta_t \sim \mathcal{N}(0, \epsilon_t), \quad (3)$$

where F is FIM or its approximation.

C. DIAGONAL APPROXIMATION

1) pSGLD

One of the first papers that propose the idea of using the adaptive preconditioning from SGD methods and applying it to improve SGLD efficacy was [27]. It follows the same algorithm to form the preconditioner as in RMSProp, where the preconditioner is updated sequentially using only the current gradient information to give a diagonal matrix estimation. This method is referred to as pSGLD.

The preconditioner matrix F is defined sequentially as follows:

$$V(\boldsymbol{\theta}_{t+1}) = \alpha V(\boldsymbol{\theta}_t) + (1-\alpha)\bar{g}(\boldsymbol{\theta}_t; D^t) \odot \bar{g}(\boldsymbol{\theta}_t; D^t)\,, \quad (4)$$
$$F(\boldsymbol{\theta}_{t+1}) = \text{diag}\left(\mathbf{1} \oslash \left(\lambda \mathbf{1} + \sqrt{V(\boldsymbol{\theta}_{t+1})}\right)\right)\,, \quad (5)$$

where $\bar{g}(\boldsymbol{\theta}_t; D^t)$ is the sample mean of the gradient using minibatch D^t, and $\alpha \in [0, 1]$. Operators $\odot$ and $\oslash$ represent element-wise matrix product and division, respectively.

2) ASGLD

ASGLD uses a diagonal approximation matrix to precondition the noise term of SGLD [6]. The preconditioner is based on a diagonal approximation of the second-order moment of gradient updates, inspired by the method of adding momentum to SGLD in SGHMC [9]. Different from other preconditioning methods included in this paper, which apply preconditioning to both the gradient and the noise term, ASGLD only applies preconditioning to the noise term.

The preconditioner matrix F is defined sequentially as follows:

$$\mu_t = \rho\mu_{t-1} + (1-\rho)\bar{g}(\boldsymbol{\theta}_t)\,, \quad (6)$$
$$F_t = \rho F_{t-1} + (1-\rho)(\bar{g}(\boldsymbol{\theta}_t) - \mu_t)(\bar{g}(\boldsymbol{\theta}_t) - \mu_{t-1})\,, \quad (7)$$

where ρ is an additional hyperparameter for momentum. There is also hyperparameter ψ which controls the amount of noise to be injected after preconditioning:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \epsilon_t(\bar{g}(\boldsymbol{\theta}_t) + \psi\eta_t), \quad \eta_t \sim \mathcal{N}(\mu_t, C_t) \quad (8)$$

D. BLOCK-DIAGONAL APPROXIMATION

1) K-FAC

The first approximation made for FIM consists of treating each layer of the DNN separately while ignoring cross-layer terms. This results in a first block-diagonal approximation of F where each block $F^{(l)}$ only considers the parameters of a single layer l. Typically, $F^{(l)}$ can still be very large. An alternative technique from [28] proposes to approximate $F^{(l)}$ as a Kronecker product of two smaller matrices so that $F^{(l)} \approx A \otimes B$. This is much cheaper to store, compute, and invert because $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$. Specifically, for a layer l that receives input of size d_{in} and computes output of size d_{out}, the Kronecker-factored approximation of corresponding $F^{(l)}$ would be two matrices of size $d_{in} \times d_{in}$ and $d_{out} \times d_{out}$, whereas the full $F^{(l)}$ would be of size $d_{in}d_{out} \times d_{in}d_{out}$ [14].

2) EKFAC

The K-FAC approximates $F^{(l)} \approx A \otimes B$ and yields the update rule: $\theta \leftarrow \theta - \eta(A \otimes B)^{-1}\nabla_\theta$. The eigen decomposition of the Kronecker product $A \otimes B$ of two real symmetric positive semi-definite matrices can be expressed using their own eigen decomposition $A = U_A S_A U_A^T$ and $B = U_B S_B U_B^T$, yielding $A \otimes B = (U_A S_A U_A^T) \otimes (U_B S_B U_B^T) = (U_A \otimes U_B)(S_A \otimes S_B)(U_A \otimes U_B)^T$. $U_A \otimes U_B$ gives the orthogonal eigen basis of the Kronecker product, and $S_A \otimes S_B$ is the diagonal matrix containing the associated eigen values. This can be interpreted as K-FAC uses $U_A \otimes U_B$ directions to approximate FIM eigen vectors U and utilizes approximate scaling $S_A \otimes S_B$.

EKFAC proposed to correct the scaling of K-FAC by replacing $U_A \otimes U_B$ with diagonal matrix defined by $S^*_{ii} = s^*_i = \mathbb{E}[((U_A \otimes U_B)^T \nabla_\theta)^2_i]$, where s^* is a vector of second moments of the gradient vector coordinates in the approximate basis $U_A \otimes U_B$. Reference [14] proved that S^* is the optimal diagonal rescaling in that basis such that we will always have $\|F - F_{EKFAC}\|_F \leq \|F - F_{K-FAC}\|_F$, where $\|\cdot\|_F$ denotes the Frobenius norm.

III. EKFAC PRECONDITIONED SGLD

EKFAC is proven to give a more accurate approximation of the FIM compared to K-FAC. Although not guaranteed, this potentially leads to better parameter updates when applied in training the DNN model [14]. Furthermore, Nado et al. showed that applying K-FAC preconditioning can improve SGLD more than using a preconditioner based on diagonal approximation as in pSGLD [29]. Hence, it is quite reasonable to implement the EKFAC preconditioning for SGLD. We hypothesize that it could improve the SGLD performance even more. Therefore, we perform numerical experiments to validate that hypothesis.

EKSGLD works by estimating preconditioner F^{-1} in (3) using EKFAC. We present high-level pseudocode for the proposed method in algorithm 1. It is almost identical to the

pseudocode for EKFAC presented in [14], with additional parameters and procedures shown in blue. Basically, we add a preconditioned Gaussian noise from SGLD during the parameter update of EKFAC once the training has passed the burn-in phase.

Algorithm 1 EKSGLD

Require: m: recompute eigenbasis every m minibatches
Require: ϵ: learning rate
Require: ν: damping parameter
Require: b: burn-in steps

procedure EKFAC(D_{train})
  while convergence is not reached, iteration i **do**
    Sample a minibatch D from D_{train}
    Do forward and backprop pass to obtain h and δ
    for all layer l **do**
      if $i \,\%\, m = 0$ **then**
        ComputeEigenBasis(D, l)
      end if
      ComputeScalings(D, l)
      $\nabla^{\text{mini}} \leftarrow \mathbb{E}_{(x,y)\in D}\left[\nabla_\theta^{(l)}(x, y)\right]$
      UpdateParameters(∇^{mini}, l)
    end for
  end while
end procedure

procedure ComputeEigenBasis(D, l)
  $U_A^{(l)}, S_A^{(l)} \leftarrow \text{eigendecomposition}\left(\mathbb{E}_D\left[h^{(l)}h^{(l)\top}\right]\right)$
  $U_B^{(l)}, S_B^{(l)} \leftarrow \text{eigendecomposition}\left(\mathbb{E}_D\left[\delta^{(l)}\delta^{(l)\top}\right]\right)$
end procedure

procedure ComputeScalings(D, l)
  $s^{*(l)} \leftarrow \mathbb{E}_D\left[\left(\left(U_A^{(l)} \otimes U_B^{(l)}\right)^\top \nabla_\theta^{(l)}\right)^2\right]$
end procedure

function Precondition(M, l)
  $\tilde{M} \leftarrow \left(U_A^{(l)} \otimes U_B^{(l)}\right)^\top M$
  $\tilde{M} \leftarrow \tilde{M}/s^{*(l)} + \nu$
  $M^{\text{precond}} \leftarrow \left(U_A^{(l)} \otimes U_B^{(l)}\right)\tilde{M}$
  return M^{precond}
end function

procedure UpdateParameters(∇^{mini}, l)
  $\nabla^{\text{precond}} \leftarrow \text{Precondition}(\nabla^{\text{mini}}, l)$
  if $i > b$ **then**
    $\eta^{\text{precond}} \leftarrow \text{Precondition}(\mathcal{N}(0, \epsilon)\,, l)$
    $\theta^{(l)} \leftarrow \theta^{(l)} - \epsilon\nabla^{\text{precond}} - \eta^{\text{precond}}$
  else
    $\theta^{(l)} \leftarrow \theta^{(l)} - \epsilon\nabla^{\text{precond}}$
  end if
end procedure

IV. METRICS AND METHODS

This section describes, in brief, the metrics used in this work. We use arrows to indicate which direction is better.

Accuracy ↑: Multiclass classification is a task where the input is to be classified into one, and only one, of l nonoverlapping classes. One of the most basic and standard metrics for classification is accuracy, which measures the overall effectiveness of a classifier [39].

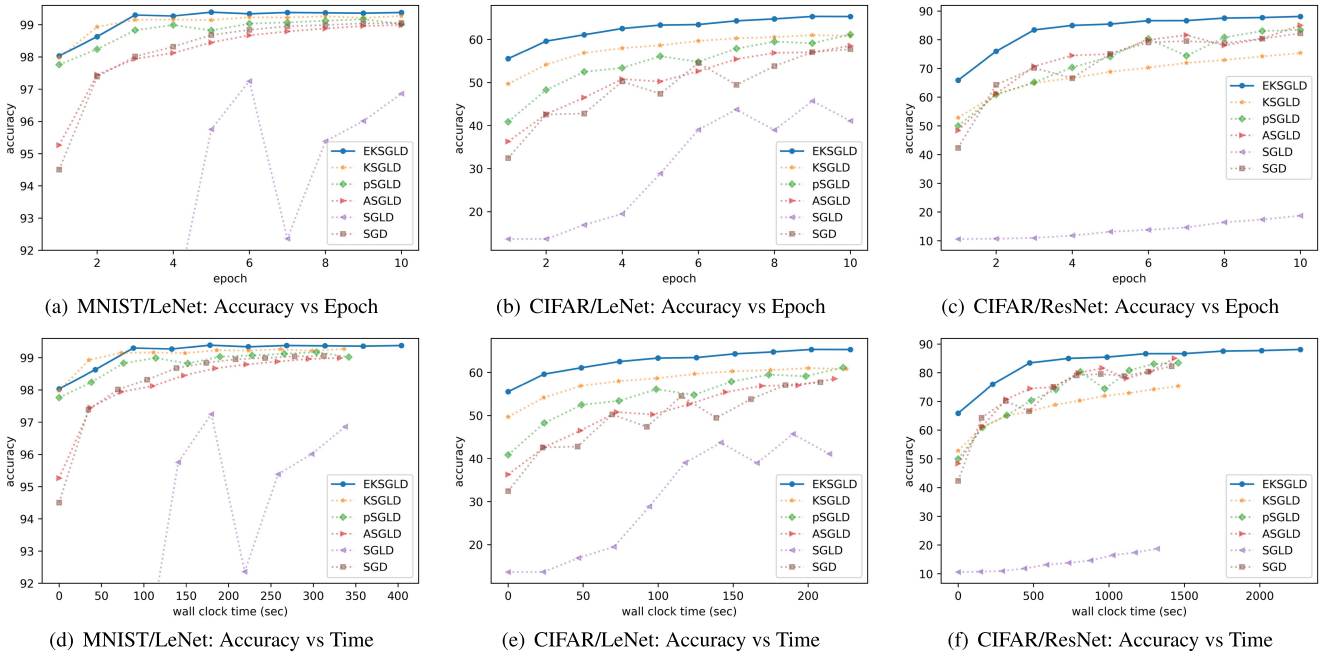
AUC$_\mu$ ↑: The area under the receiver operating characteristic curve, also known as the AUC, has been used for measuring classifier performance everywhere in machine learning research. This metric is initially defined for binary classification, with only two target classes. Here, we use AUC$_\mu$, a recently proposed extension of AUC for multiclass classification, which has similar computational complexity to AUC and maintains the properties of AUC for similar interpretation and uses [22].

NLL ↓: The negative log-likelihood (NLL) as a loss function comes from a probabilistic formulation of the learning problem regarding the maximum conditional probability principle. Given dataset D, we must find the parameter value that maximizes the conditional probability of all the labels given all the inputs in the dataset [26]. Besides being a loss function in training neural network models, NLL is also a common metric for evaluating the quality of model uncertainty on some held-out sets.

ECE ↓ and **MCE** ↓: Expected calibration error (ECE) measures confidence calibration quality relative to the ideal condition where confidence matches empirical accuracy exactly. The predictions are sorted and partitioned into K fixed number of bins in computing this measure. We use the default of $K = 15$ for all calibration measurements that use binning throughout the experiments. Maximum calibration error (MCE) is similar to ECE, but instead of calculating expectations over the bins, MCE only considers maximum error among the bins [30]. Despite receiving criticism recently [33], ECE remains the most popular metric used for measuring calibration in recent publications.

OE ↓: High confidence but incorrect forecasts can be extremely devastating in high-risk applications. Overconfidence error (OE) is a variant of ECE in which predictions are only penalized when confidence surpasses accuracy [41].

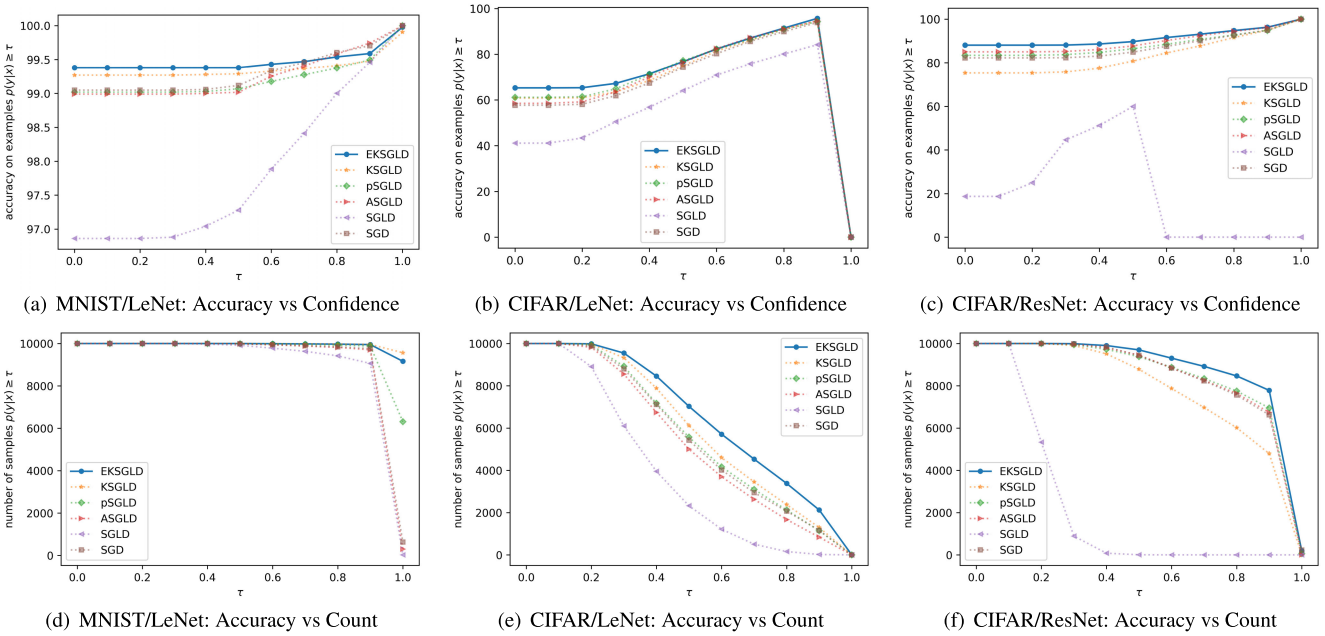
SCE ↓ and **TACE** ↓: These measurements are provided in an attempt to solve the shortcomings of ECE. Adaptive calibration error (ACE) calculates the final error score by employing adaptive bin intervals that split the data into equal numbers of predictions in each bin rather than equal bin intervals as in ECE. Thresholded adaptive calibration error (TACE) aims to improve ACE's efficiency, particularly in several target classes, by calculating the calibration error score using only predictions over a predefined threshold. We set the threshold to **0.001** when computing TACE in this experiment. ECE is ideal for binary classification since it focuses exclusively on the likelihood of the class with the highest probability for every given data point. Static



(a) MNIST/LeNet: Accuracy vs Epoch (b) CIFAR/LeNet: Accuracy vs Epoch (c) CIFAR/ResNet: Accuracy vs Epoch

(d) MNIST/LeNet: Accuracy vs Time (e) CIFAR/LeNet: Accuracy vs Time (f) CIFAR/ResNet: Accuracy vs Time

FIGURE 1. Comparing classification performance based on validation accuracy of different SGLD algorithms over training epoch (top) and over training time (bottom). EKSGLD shows the highest accuracy after training with the same number of epochs in all of the experiments: (a), (b), and (c). Training with EKSGLD requires more computation, indicated by the longer overall training time. However, it still shows the highest accuracy when compared to the other methods at any given wall clock time, which means that we could potentially train EKSGLD with less number of epochs and still get better or comparable accuracy: (d), (e), and (f).



(a) MNIST/LeNet: Accuracy vs Confidence (b) CIFAR/LeNet: Accuracy vs Confidence (c) CIFAR/ResNet: Accuracy vs Confidence

(d) MNIST/LeNet: Accuracy vs Count (e) CIFAR/LeNet: Accuracy vs Count (f) CIFAR/ResNet: Accuracy vs Count

FIGURE 2. Comparing the accuracy (top) and the number of predictions (bottom) when we only consider the predictions with confidence equal to or above the threshold τ. All methods show increasing accuracy as the threshold τ increases, which is expected: (a), (b), and (c). EKSGLD consistently retains the highest number of samples in all of the experiments except in MNIST/LeNet with $\tau = 1$, where KSGLD has the highest number of samples: (d), (e), and (f).

calibration error (SCE) is a straightforward modification of ECE that considers the likelihood of each class for each given data point in a multiclass environment. [33].

We present accuracy and AUC_μ as the primary metrics for assessing classification performance in the experiment using the validation set. We provide and evaluate six metrics

TABLE 2. Classification performance and confidence calibration of different SGLD algorithms on MNIST and CIFAR-10 validation set. No single method performed best across all metrics and experiments, but EKSGLD achieved the best score (marked with a bold number) most often.

Dataset/Model	Optimizer	ECE	MCE	NLL	SCE	TACE	OE	Accuracy	AUCμ	Time/epoch
CIFAR10/ResNet-18	EKSGLD	0.04121	0.254	**0.376**	0.8846	**0.0553**	0.0344	**88.08**	**0.99415**	241.02 +/- 11.39
CIFAR10/ResNet-18	KSGLD	0.04874	0.801	0.714	0.8601	0.1667	0.0359	75.34	0.97827	153.22 +/- 2.28
CIFAR10/ResNet-18	pSGLD	0.05304	0.167	0.530	0.8797	0.1006	0.0442	83.41	0.99205	152.08 +/- 3.68
CIFAR10/ResNet-18	ASGLD	**0.03120**	**0.091**	0.449	0.8782	0.0966	0.0260	85.02	0.99168	150.51 +/- 0.29
CIFAR10/ResNet-18	SGLD	0.04798	0.648	2.214	**0.7417**	0.7408	**0.0076**	18.72	0.80031	**138.12 +/- 0.89**
CIFAR10/ResNet-18	SGD	0.05614	0.188	0.551	0.8787	0.0955	0.0452	82.25	0.99030	148.95 +/- 3.70
CIFAR10/LeNet-5	EKSGLD	**0.00919**	**0.025**	**0.978**	0.8338	**0.3137**	0.0051	**65.30**	**0.95980**	22.48 +/- 0.73
CIFAR10/LeNet-5	KSGLD	0.01377	0.028	1.104	0.8242	0.3656	0.0017	60.82	0.94942	22.25 +/- 0.56
CIFAR10/LeNet-5	pSGLD	0.04163	0.071	1.117	0.8244	0.4082	0.0010	61.15	0.95060	21.98 +/- 0.57
CIFAR10/LeNet-5	ASGLD	0.04555	0.077	1.190	0.8200	0.4578	**0.0001**	58.52	0.94616	21.54 +/- 0.06
CIFAR10/LeNet-5	SGLD	0.03175	0.152	1.616	**0.7854**	0.6182	0.0009	41.08	0.89316	21.07 +/- 0.25
CIFAR10/LeNet-5	SGD	0.01897	0.062	1.188	0.8197	0.4203	0.0027	57.72	0.94469	**20.39 +/- 0.06**
MNIST/LeNet-4	EKSGLD	0.00449	0.374	0.036	0.8997	**0.0003**	0.0042	**99.38**	**0.99998**	40.65 +/- 1.05
MNIST/LeNet-4	KSGLD	0.00516	0.617	0.062	**0.8892**	0.0004	0.0049	99.27	0.99997	33.21 +/- 0.79
MNIST/LeNet-4	pSGLD	0.00591	0.415	0.042	0.8995	0.0017	0.0054	99.02	0.99996	32.56 +/- 0.07
MNIST/LeNet-4	ASGLD	0.00302	**0.349**	0.031	0.8987	0.0100	**0.0008**	98.99	0.99996	33.50 +/- 0.66
MNIST/LeNet-4	SGLD	0.00523	0.688	0.096	0.8952	0.0337	0.0010	96.86	0.99961	33.05 +/- 2.33
MNIST/LeNet-4	SGD	**0.00233**	0.381	**0.028**	0.8990	0.0076	0.0010	99.05	0.99996	**30.35 +/- 0.54**

for assessing the quality of model uncertainty: ECE, MCE, NLL, SCE, TACE, and OE scores. Additionally, we offer the mean and standard deviation of training length in each epoch to compare the computing resources consumed by each optimizer.

We evaluate the accuracy and ECE on rotated pictures from the validation set at various rotational degrees in the experiment under dataset shift. This is intended to illustrate how classification performance changes and if prediction uncertainty can be maintained while shifting intensity on the test dataset changes.

We did not assess accuracy in the experiment with entirely OOD data since the train data had a completely different set of class labels than the test data [35]. We provide histograms of predicted entropy for OOD data and compare them to predictive entropy for i.i.d. data. On OOD data, we anticipate a considerably greater predictive entropy. Additionally, we give the number of samples with a confidence score greater than a specified confidence level τ. We should anticipate a poor confidence score for all predictions made using OOD data, as the test data are completely unrelated to the train data.

V. EXPERIMENTS AND RESULTS

We evaluate the performance and the predictive uncertainty quality of DNN models on MNIST and CIFAR-10 datasets. For training the MNIST dataset, we use a four-layer LeNet architecture following Palacci and Hess in [36] and refer to it as LeNet-4 for brevity. Despite having less number of layers, LeNet-4 contains more learnable parameters than the more commonly used LeNet-5 owing to the larger number of output channels in its convolutional layers. Table 3 presents the detailed architecture for LeNet-4. We do not include detailed architecture for LeNet-5 and ResNet-18 since they follow standard settings commonly used in other machine learning literature.

TABLE 3. LeNet-4 architecture. This model contains 909,770 learnable parameters compared to LeNet-5, which only contains 62,006.

Layer	Output shape	# Parameters
Conv2D	64 x 28 x 28	1664
Conv2D	64 x 14 x 14	102464
Linear	256	803072
Linear	10	2570

For training on the CIFAR-10 dataset, we use two models of different capacities, namely, 5-layer LeNet and 18-layer ResNet neural architectures, following Osawa et al. in [34]. We train the models using SGLD with different preconditioning algorithms, including vanilla SGLD and SGD as a baseline. We follow standard training and testing protocols for each dataset, model, and optimization algorithm. However, we additionally evaluate results on increasingly shifted data and OOD dataset, loosely following the procedure in [35]. For reproducibility purposes, our PyTorch codes are available online at https://github.com/har07/ngld-calibration/.

A. HYPERPARAMETERS

Table 4 summarizes the hyperparameter configurations used in both MNIST and CIFAR-10 experiments. For each method, we referred to existing literature to set the initial hyperparameter configuration, then searched around the initial configuration and took the best hyperparameter configuration based on training accuracy. Note that the chosen hyperparameter configurations might not be optimal since we did not have the required computing power to do extensive hyperparameter tuning over a wide range of values and combinations. All experiments in this paper were run in the free Google Colaboratory service environment. Hence, it is supposed to require moderately low compute power and GPU

TABLE 4. Hyperparameter configuration of every optimizer used in the experiments.

Optimizer	Hyperparameter	
EKSGLD	initial learning rate	0.005
	burn-in steps	600
	damping parameter	0.001
	update frequency	50
KSGLD	initial learning rate	0.032
	burn-in steps	600
	damping parameter	0.001
	update frequency	50
pSGLD	initial learning rate	0.001
	burn-in steps	300
	running average parameter	0.95
ASGLD	initial learning rate	0.1
	momentum	0.9
	weight decay	0.0005
	noise parameter	0.01
SGLD	initial learning rate	0.15
	burn-in steps	300
SGD	initial learning rate	0.1

memory space, which opens up possibilities for those with limited resources to reproduce or build upon this benchmark.

For SGD, SGLD, and pSGLD, the initial hyperparameter configuration is based on the MNIST classification experiment from [27]. For ASGLD, we referred to the hyperparameter configuration in the CIFAR-10 classification experiment from [6]. Finally, for KSGLD and EKSGLD, we referred to the MNIST and CIFAR-10 classification experiments from [14]. We train the same model architecture for all methods with a minibatch size of 200. We trained the model for ten epochs in the MNIST classification experiment and for 50 epochs in the CIFAR-10 classification experiment, with the learning rate decreasing by half after every 20 epochs, loosely following the block decay learning rate schedule in [27].

B. MODEL ACCURACY

We train the same model architecture using various optimization techniques and assess the accuracy at each epoch using a standard validation set. As shown in Fig. 1, EKSGLD consistently achieves the highest accuracy throughout all three experiments: MNIST/LeNet, CIFAR/LeNet, and CIFAR/ResNet. Methods such as EKSGLD and KSGLD that use a block-diagonal approximation of FIM are known to be computationally more costly than those that use diagonal approximation. As seen in the bottom row of Fig. 1, EKSGLD took much longer to complete the ten training epochs, particularly in the experiment using ResNet-18, which has deeper layers and a more significant number of parameters. However, the exact figure demonstrates that the accuracy of EKSGLD is equivalent to or greater than that of the other approaches after the same training time. Other figures and tables in this paper are obtained from models that were trained for the same number of epochs, not the same amount of training time.

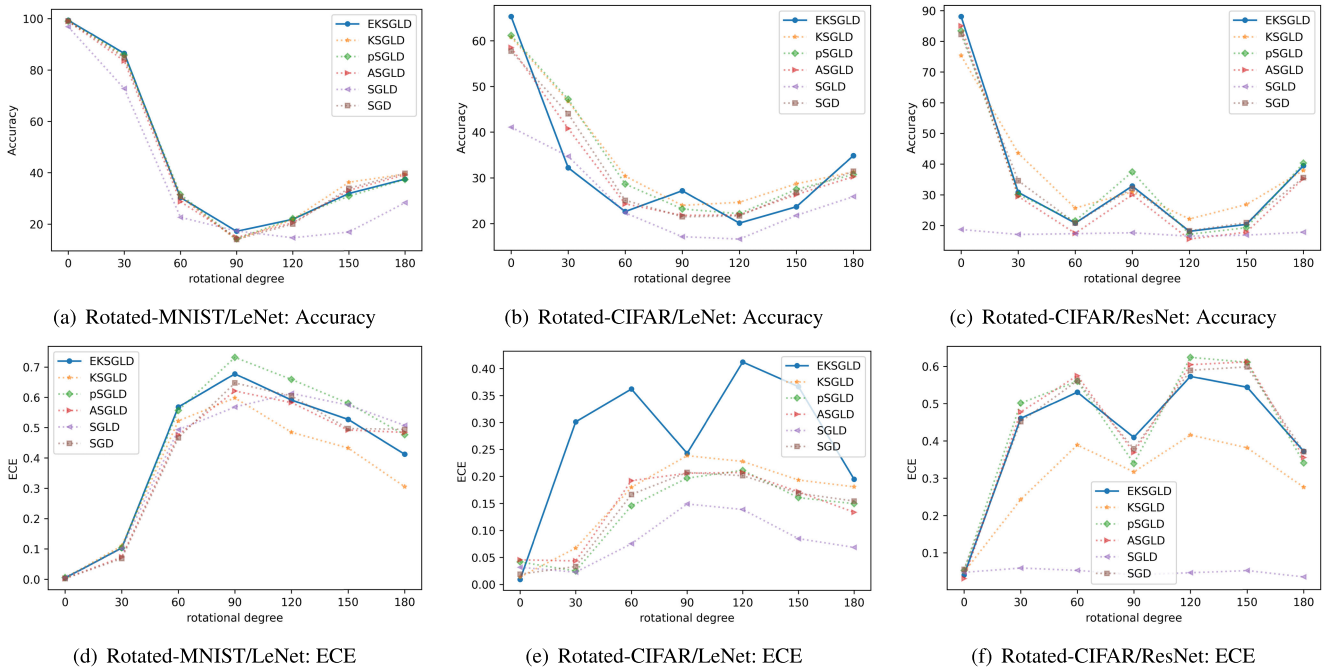
KSGLD was the second-best approach in both the MNIST and CIFAR trials that used the LeNet neural architecture. However, its performance decreased dramatically in the CIFAR experiment that used the ResNet neural architecture. pSGLD and ASGLD showed a more consistent performance across all experiments than KSGLD. Additionally, we see that training using SGLD is unstable and particularly difficult to optimize in the experiment on CIFAR utilizing the ResNet architecture. Due to the low precision of SGLD hyperparameters, we attempted to tune them for a longer period of time than the other approaches, but we were unable to discover hyperparameters' values that resulted in comparable performance even after the extended tuning time.

From now on, we will make predictions using a mixture of 10 models created after each training period for all sorts of SGLD approaches. We employ just one model from the end of the previous training period for SGD. In Table 2, we calculate classification performance and prediction uncertainty quality parameters. As shown, EKSGLD continues to outperform the other approaches in terms of accuracy and AUCμ metrics, but its average training time is significantly greater. Generally, the relative ordering of methods based on accuracy almost always matches the ordering based on AUCμ, except for the second and third positions in the CIFAR-10 experiment using ResNet architecture, where pSGLD and ASGLD switch positions depending on whether the order is determined by accuracy or AUCμ. The following section examines the remaining metrics in the table that relate to the quality of prediction uncertainty.

C. PREDICTIVE UNCERTAINTY QUALITY

We now investigate models' predictive uncertainty quality using the same set of neural networks, optimization techniques, and picture datasets. This section begins by assessing the predictive distribution on the i.i.d. dataset. We utilize a validation set that contains data drawn from the same distribution as the training set. Table 2 demonstrates that no single strategy consistently outperformed the others across all experiments and measurements. The table contains 18 figures describing the prediction uncertainty associated with each optimizer, especially six metrics (ECE, MCE, NLL, SCE, TACE, and OE) across three trials (MNIST, CIFAR/LeNet, and CIFAR/ResNet). As shown, EKSGLD was the best most frequently, precisely on seven of the 18 instances, followed by ASGLD on five occasions.

Additionally, we investigate the effect of introducing a predictive confidence threshold τ on the model's accuracy. We anticipate that accuracy will rise as the value of τ is increased, or in other words, as more predictions with low confidence scores are discarded. Fig. 2 shows this is the case for all methods. The ranking of techniques by accuracy is nearly constant throughout all τ values in all trials, with EKSGLD being on the top, except for the MNIST experiment (Fig. 2(a)), where the accuracy of SGD and ASGLD begins



(a) Rotated-MNIST/LeNet: Accuracy (b) Rotated-CIFAR/LeNet: Accuracy (c) Rotated-CIFAR/ResNet: Accuracy

(d) Rotated-MNIST/LeNet: ECE (e) Rotated-CIFAR/LeNet: ECE (f) Rotated-CIFAR/ResNet: ECE

FIGURE 3. Comparing validation accuracy and calibration of different SGLD algorithms under dataset shift. All methods experience degradation in accuracy under dataset shift following similar graph patterns: (a), (b), and (c). KSGLD is slightly more robust, showing low ECE scores on (d) and (f). However, both KSGLD and EKSGLD, i.e., the two methods based on block-diagonal approximations, have high calibration errors based on ECE in the experiment using minimal model parameters (e).

to rise faster than that of KSGLD and EKSGLD at $\tau \geq 0.6$. However, when the number of samples is considered (Fig. 2(d)), KSGLD and EKSGLD have somewhat more confident forecasts compared to SGD and ASGLD.

D. PREDICTIVE UNCERTAINTY UNDER DATASET SHIFT

In this part, we continue to evaluate the predictive distribution's quality by rotating the image from MNIST and CIFAR-10 in various degrees to mimic distributional shifts with varying intensities. While it is predicted that model performance will deteriorate as the magnitude of the distributional shift rises, it would be ideal if the model could maintain its predictive distribution quality.

In practice, we may apply a confidence threshold to the predictions to improve model accuracy by eliminating predictions with low confidence, which, when the model predictions are well-calibrated, means eliminating forecasts with a lesser probability of being right. Regrettably, the outcome indicates that this is not the case. Fig. 3 demonstrates that when a distributional shift is added, both accuracy and prediction uncertainty quality decline for all techniques.

Starting with comparable accuracy in the validation set, SGLD accuracy degrades more rapidly than the others in the MNIST distributional shift experiment. EKSGLD was likewise unable to maintain the top position in terms of accuracy when the distributional change occurred. KSGLD consistently maintains a slightly greater accuracy throughout all of the distributional shift studies.

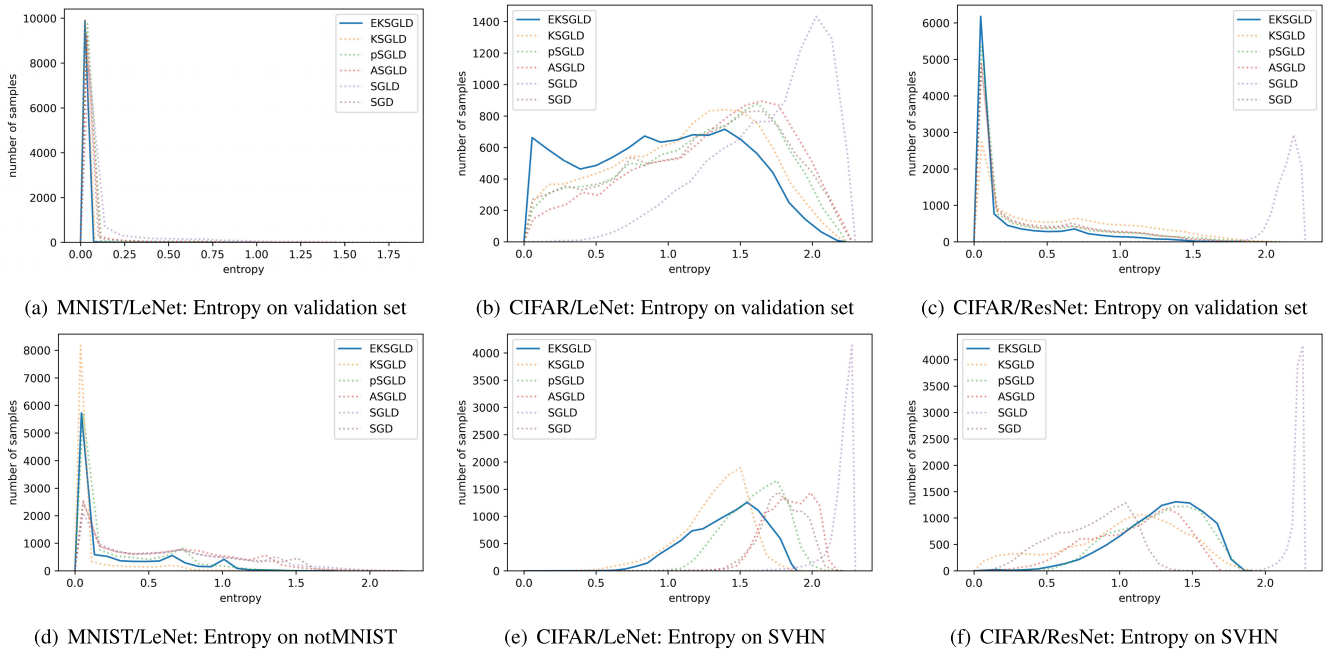
Except for 90° rotation, EKSGLD exhibits a much greater ECE when the CIFAR dataset is rotated using the LeNet architecture. In the other two studies, namely, rotated MNIST and rotated CIFAR with ResNet architecture, pSGLD achieved the greatest ECE scores. In rotated MNIST and rotated CIFAR with ResNet, KSGLD gets the lowest ECE scores.

Generally, KSGLD is slightly more resilient to distributional change than the other approaches. However, our CIFAR/LeNet experiment demonstrates that approaches based on second-order approximation are prone to substantial calibration errors under distributional shifts when employed with an under-parameterized model.

E. PREDICTIVE UNCERTAINTY ON OOD

The trials' final section assesses models' prediction uncertainty using totally OOD data. We assess models trained on MNIST using the notMNIST dataset [8] and models trained on CIFAR using the SVHN dataset [32]. Each pair of datasets comprises a distinct set of labels that do not overlap, and we consider OOD data to lack ground-truth labels. One may imagine that predictive distributions on OOD data would have a high degree of entropy, whereas predictive distributions on i.i.d. data would have a low degree of entropy. The difference in the entropy of prediction distributions can be used by the model to signify what it knows or does not know.

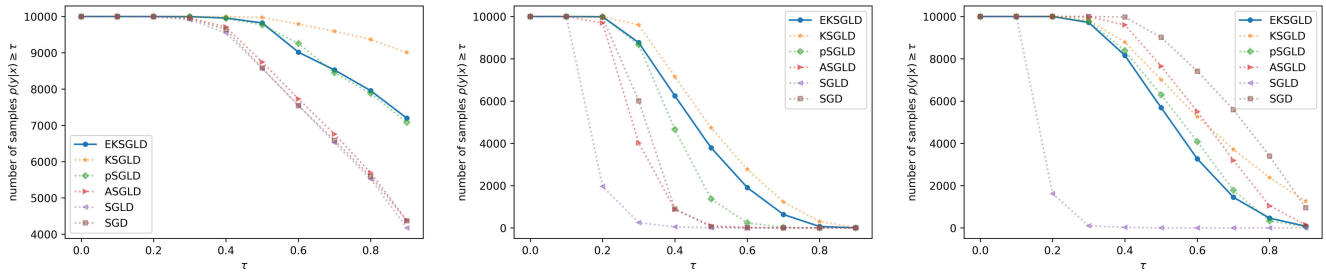
We provide the entropy histogram for each technique on OOD data and compare it to the entropy on the validation set, which is assumed to be i.i.d. data, as shown in Fig. 4.



(a) MNIST/LeNet: Entropy on validation set (b) CIFAR/LeNet: Entropy on validation set (c) CIFAR/ResNet: Entropy on validation set

(d) MNIST/LeNet: Entropy on notMNIST (e) CIFAR/LeNet: Entropy on SVHN (f) CIFAR/ResNet: Entropy on SVHN

FIGURE 4. Comparing predictive entropy histogram of different SGLD algorithms on i.i.d. (top) and OOD (bottom) datasets. Overall, the entropy on the OOD is relatively higher compared to the entropy on the validation set.



(a) notMNIST/LeNet: Count vs Confidence (b) SVHN/LeNet: Count vs Confidence (c) SVHN/ResNet: Count vs Confidence

FIGURE 5. Comparing the number of samples of different SGLD algorithms on the OOD dataset for different thresholds τ. SGLD could be the most robust method showing the lowest number of samples with high confidence scores; however, previously, we see that this is correlated with its low accuracy on the validation set. KSGLD is shown to be the most overconfident in (a) and (b), whereas SGD is the most overconfident in (c).

Combining the observations from Table 2 (accuracy) and Fig. 4 (entropy histogram), we see that the predictive distributions on the validation set have a relatively higher entropy when the model accuracy is low, and a relatively low entropy when the model accuracy is high, as illustrated in Fig. 4. For example, SGLD has the lowest accuracy on CIFAR-10 experiments according to Table 2, and it has the highest entropy according to Fig. 4 (b) and (c). Additionally, for all approaches, we note that the predictive distributions on OOD data have a larger entropy than the respective predictive distributions on the validation set, which is consistent with expectations.

As shown in Fig. 5, vanilla SGLD is the most resilient approach to OOD data, with a low confidence score for all of its predictions, particularly in the two trials utilizing the SVHN dataset. However, as previously seen in Fig. 2 and Table 2, SGLD likewise exhibits poor confidence and accuracy on i.i.d. data. SGD and ASGLD are more resilient in the two tests with LeNet neural architecture, but EKSGLD is more robust in the experiments involving ResNet neural architecture. Overall, no one technique consistently produces both low confidence and high accuracy predictions on OOD data and high confidence and high accuracy predictions on i.i.d. data across all tests.

F. DISCUSSIONS

We can see in Table 2 that EKSGLD achieves the best classification performance based on both accuracy and AUC_μ while also maintaining a good calibration performance based on the majority of calibration metrics, especially TACE and NLL. Ashukha et al. argued that TACE and NLL are better metrics than ECE for comparing predictive uncertainty quality [5]. Based on this argument, we can say that EKSGLD

produces the best predictive uncertainty quality on i.i.d. data. Moreover, we have illustrated the *accuracy versus confidence* curves in Fig. 2. Despite showing an expected trend where the accuracy increases as the threshold τ increases, it is still far from perfect calibration (a perfect diagonal line).

Regarding experiments on predictive uncertainty under dataset shift, we expect that accuracy decreases as shift intensity increases, but the ECE should be stable. However, we observe from Fig. 3 that calibration error (i.e., the ECE) increases as the accuracy decreases. This means that despite some methods being better than others, all methods in these experiments are not entirely robust to dataset shift.

In Fig. 4, we see a similar pattern for all methods. Generally, the entropy on OOD data is relatively higher than the entropy on the validation set. This means the models produce relatively uncertain predictions on OOD data compared to the predictions produced on i.i.d. data, which is the expected behavior. Our results from both predictive uncertainty under dataset shift and on OOD data experiments complement the results from previous papers since they do not include approximate NGLD [21], [35]. However, the takeaways are still aligned: improved performance and confidence calibration on the validation set may not always equate to the same case when the dataset is shifted and when given OOD data input.

VI. CONCLUSION

In this work, we presented EKSGLD, a new approach based on SGLD, to obtain an accurate and calibrated classification model. We show that the approach produces better accuracy and predictive uncertainty quality on i.i.d. data compared to the other tested methods, which is a step closer to a well-calibrated model. Subsequent experiments showed that maintaining predictive uncertainty quality under dataset shift and on OOD data remains challenging for all of the SGLD preconditioning approaches shown here, necessitating a possible area of future research.

It will be interesting to see how these methods perform more challenging tasks such as medical image analysis and whether simple improvements from recent works such as cyclic learning schedule [47] can improve approximate NGLD methods.

ACKNOWLEDGMENT

The authors greatly appreciated the Data Science Centre and the Directorate of Research and Development Universitas Indonesia for supporting and furthering their research.

REFERENCES

[1] A. A. Abdullah, M. M. Hassan, and Y. T. Mustafa, "A review on Bayesian deep learning in healthcare: Applications and challenges," *IEEE Access*, vol. 10, pp. 36538–36562, 2022.

[2] A. Akbari and R. Jafari, "Personalizing activity recognition models through quantifying different types of uncertainty using wearable sensors," *IEEE Trans. Biomed. Eng.*, vol. 67, no. 9, pp. 2530–2541, Sep. 2020.

[3] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, no. 2, pp. 251–276, Feb. 1998.

[4] A. O. Aseeri, "Uncertainty-aware deep learning-based cardiac arrhythmias classification model of electrocardiogram signals," *Computers*, vol. 10, no. 6, p. 82, Jun. 2021.

[5] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," 2020, *arXiv:2002.06470*.

[6] C. A. Bhardwaj, "Adaptively preconditioned stochastic gradient Langevin dynamics," in *Proc. 36th Int. Conf. Mach. Learn.*, Long Beach, CA, USA, 2019, pp. 1–6.

[7] T. Brown, B. Mann, N. Ryder, N. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, and S. Agarwal, "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1877–1901.

[8] Y. Bulatov, "Notmnist dataset," Feb. 2023. [Online]. Available: http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html

[9] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient Hamiltonian Monte Carlo," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, P. E. Xing and T. Jebara, Eds. Bejing, China, Jun. 2014, pp. 1683–1691.

[10] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds. Montreal, QC, Canada: Curran Associates, 2014, pp. 1–9.

[11] A. Filos, S. Farquhar, A. N. Gomez, T. G. J. Rudner, Z. Kenton, L. Smith, M. Alizadeh, A. de Kroon, and Y. Gal, "A systematic comparison of Bayesian deep learning robustness in diabetic retinopathy tasks," 2019, *arXiv:1912.10481*.

[12] G. Franchi, A. Bursuc, E. Aldea, S. Dubuisson, and I. Bloch, "One versus all for deep neural network for uncertainty (OVNNI) quantification," *IEEE Access*, vol. 10, pp. 7300–7312, 2022.

[13] W. Fruehwirt, A. D. Cobb, M. Mairhofer, L. Weydemann, H. Garn, R. Schmidt, T. Benke, P. Dal-Bianco, G. Ransmayr, M. Waser, D. Grossegger, P. Zhang, G. Dorffner, and S. Roberts, "Bayesian deep neural networks for low-cost neurophysiological markers of Alzheimer's disease severity," in *Proc. Mach. Learn. Health (MLH) Workshop*, 2018, pp. 1–6.

[14] T. George, C. Laurent, X. Bouthillier, N. Ballas, and P. Vincent, "Fast approximate natural gradient descent in a Kronecker factored eigenbasis," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Montreal, QC, Canada: Curran Associates, 2018, pp. 1–11.

[15] M. Gour and S. Jain, "Uncertainty-aware convolutional neural network for COVID-19 X-ray images classification," *Comput. Biol. Med.*, vol. 140, Jan. 2022, Art. no. 105047.

[16] R. Grosse and J. Martens, "A Kronecker-factored approximate Fisher matrix for convolution layers," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 573–582.

[17] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1321–1330.

[18] F. K. Gustafsson, M. Danelljan, and T. B. Schon, "Evaluating scalable Bayesian deep learning methods for robust computer vision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 1289–1298.

[19] E. T. Hastuti, A. Bustamam, P. Anki, R. Amalia, and A. Salma, "Performance of true transfer learning using CNN DenseNet121 for COVID-19 detection from chest X-ray images," in *Proc. IEEE Int. Conf. Health, Instrum. Meas., Natural Sci. (InHeNce)*, Jul. 2021, pp. 1–5.

[20] Q. Hua, Y. Yaqin, B. Wan, B. Chen, Y. Zhong, and J. Pan, "An interpretable model for ECG data based on Bayesian neural networks," *IEEE Access*, vol. 9, pp. 57001–57009, 2021.

[21] P. Izmailov, S. Vikram, M. D. Hoffman, and A. G. G. Wilson, "What are Bayesian neural network posteriors really like?" in *Proc. Int. Conf. Mach. Learn.*, vol. 139, 2021, pp. 4629–4640.

[22] R. Kleiman and D. Page, "AUC: A performance metric for multi-class machine learning models," in *Proc. Int. Conf. Mach. Learn.*, vol. 97, 2019, pp. 3439–3447.

[23] A. Kristiadi, M. Hein, and P. Hennig, "Being Bayesian, even just a bit, fixes overconfidence in ReLU networks," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5436–5446.

[24] J. Van Landeghem, M. Blaschko, B. Anckaert, and M. Moens, "Benchmarking scalable predictive uncertainty in text classification," *IEEE Access*, vol. 10, pp. 43703–43737, 2022.

[25] A. Largent, J. De Asis-Cruz, K. Kapse, S. D. Barnett, J. Murnick, S. Basu, N. Andersen, S. Norman, N. Andescavage, and C. Limperopoulos, "Automatic brain segmentation in preterm infants with post-hemorrhagic hydrocephalus using 3D Bayesian U-Net," *Hum. Brain Mapping*, vol. 43, no. 6, pp. 1895–1916, Apr. 2022.

[26] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*. Cambridge, MA, USA: MIT Press, 2006.
[27] C. Li, C. Chen, D. Carlson, and L. Carin, "Preconditioned stochastic gradient Langevin dynamics for deep neural networks," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1788–1794.
[28] Z. Tang, F. Jiang, M. Gong, H. Li, Y. Wu, F. Yu, Z. Wang, and M. Wang, "SKFAC: Training neural networks with faster Kronecker-factored approximate curvature," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Lille, France, Jun. 2021, pp. 2408–2417.
[29] Z. Nado, J. Snoek, R. Grosse, D. Duvenaud, B. Xu, and J. Martens, "Stochastic gradient Langevin dynamics that exploit neural network structure," in *Proc. ICLR*, 2018, pp. 1–4.
[30] M. P. Naeini, F. Gregory Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using Bayesian binning," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2901–2907.
[31] R. M. Neal, *Bayesian Learning for Neural Networks*. Berlin, Germany: Springer-Verlag, 1996.
[32] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and Y. A. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS*, 2011, pp. 1–9.
[33] J. Nixon, W. M. Dusenberry, L. Zhang, G. Jerfel, and D. Tran, "Measuring calibration in deep learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops*, Jun. 2019, pp. 1–4.
[34] K. Osawa, S. Swaroop, M. E. E. Khan, A. Jain, R. Eschenhagen, R. E. Turner, and R. Yokota, "Practical deep learning with Bayesian principles," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32. Vancouver, BC, Canada: Curran Associates, 2019, pp. 1–13.
[35] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–12.
[36] H. Palacci and H. Hess, "Scalable natural gradient Langevin dynamics in practice," in *Proc. 35th Int. Conf. Mach. Learn.*, Stockholm, Sweden, 2018, pp. 1–5.
[37] C. John Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. Cambridge, MA, USA: MIT Press, 1999, pp. 61–74.
[38] J. Rodríguez-Puigvert, R. Martínez-Cantín, and J. Civera, "Bayesian deep neural networks for supervised learning of single-view depth," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2565–2572, Apr. 2022.
[39] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009.
[40] B. Song, S. Sunny, S. Li, K. Gurushanth, P. Mendonca, N. Mukhia, S. Patrick, S. Gurudath, S. Raghavan, I. Tsusennaro, and S. T. Leivon, "Bayesian deep learning for reliable oral cancer image classification," *Biomed. Opt. Exp.*, vol. 12, no. 10, pp. 6422–6430, Oct. 2021.
[41] S. Thulasidasan, G. Chennupati, J. Bilmes, T. Bhattacharya, and S. Michalak, "On mixup training: Improved calibration and predictive uncertainty for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 1–12.
[42] M. P. Vadera, J. Li, A. Cobb, B. Jalaian, T. Abdelzaher, and B. Marlin, "URSABench: A system for comprehensive benchmarking of Bayesian deep neural network models and inference methods," in *Proc. Mach. Learn. Syst.*, 2022, pp. 217–237.
[43] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. 28th Int. Conf. Mach. Learn.*, Madison, WI, USA, 2011, pp. 681–688.
[44] F. Wenzel, K. Roth, S. B. Veeling, J. Swiatkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin, "How good is the Bayes posterior in deep neural networks really?" 2020, *arXiv:2002.02405*.
[45] B. Zadrozny and C. Elkan, "Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers," in *Proc. 18th Int. Conf. Mach. Learn.*, San Francisco, CA, USA, 2001, pp. 609–616.
[46] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Jul. 2002, pp. 694–699.
[47] R. Zhang, C. Li, J. Zhang, C. Chen, and A. G. Wilson, "Cyclical stochastic gradient MCMC for Bayesian deep learning," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–28.

HANIF AMAL ROBBANI received the B.Sc. degree in computer science from IPB University, Bogor, Indonesia, in 2013. He is currently pursuing the master's degree with the Big Data and Advanced Computing Laboratory, Department of Mathematics, Universitas Indonesia. He has been a lead Product Engineer with Navcore Nextology, since 2016, where he specializes in building platforms for API automation, identity as a service, and cluster automation using C#, Python, Go, Kubernetes, and other cutting-edge technologies. His research interests include medical image analysis, Bayesian deep learning, and machine learning operations (MLOps).

ALHADI BUSTAMAM received the B.Sc. degree (Hons.) in computational mathematics, in 1996, the master's degree in computer science from Universitas Indonesia, in 2002, and the Ph.D. degree in bioinformatics from The University of Queensland, Australia, in 2011. Currently, he is a Professor and the Head of the Bioinformatics and Advanced Computing Laboratory (BACL), Department of Mathematics. He is also the Chairperson of Data Science Centre (DSC) (https://dsc.ui.ac.id), Universitas Indonesia. His research interests include high-performance computing approaches to computational mathematics, computational biology, bioinformatics, computer science, data science, and artificial intelligence.

RISMAN ADNAN received the B.Sc. and M.Sc. degrees in theoretical physics and the Ph.D. degree in computer science from Universitas Indonesia, in 1998, 2000, and 2021, respectively. From 1995 to 2000, he was a Research Assistant with the Amorphous Semiconductor Laboratory, Department of Physics, Universitas Indonesia. After completing the master's degree, he was a lead software engineer at several IT companies. In 2004, he spent ten years as the Director of the developer ecosystem with Microsoft Indonesia. In 2014, he joined Samsung Research and Development Indonesia (SRIN) as the Chief Technology Officer to incubate and nurture AI, the IoT, and cloud technology competencies. He joined as the Digital Technology Director with Kalbe Digital Lab, in 2023. His research interests include theoretical physics, machine learning, and quantum computing.

SHANDAR AHMAD received the Ph.D. degree in physics. He has been working in the field of bioinformatics since 2002. He has developed novel methods to predict various aspects of protein-DNA and other bio-molecular interactions. His major contributions are in the development of first generation DNA-binding site and binding protein prediction from sequence, investigating relationships between free energy and stability of protein-DNA complexes in terms of conserved residues networks, and classifying conformational changes on complex formation comprehensively. His current research interests include systems and genome-wide predictions of bio-molecular interactions, their role as a bio-markers of diseases and in pathways and computational diagnostics, such as uni-parental disomy. He is also working with help from experimental Biologists on determining the role of DNA shape in host pathogen interactions and determinants of pathogen, cellular, and strain specificity from a genomic perspective.