Efficient and Effective Uncertainty Quantification in Gradient Boosting via Cyclical Gradient MCMC

Tian Tan[1,*], Carlos Huertas[1] and Qi Zhao[1]

[1]Buyer Risk Prevention - ML, WW Customer Trust, Amazon, Seattle, WA 98109, USA

Abstract

Gradient boosting decision trees (GBDTs) are widely applied on tabular data in real-world ML systems. Quantifying uncertainty in GBDT models is thus essential for decision making and for avoiding costly mistakes to ensure an interpretable and safe deployment of tree-based models. Recently, Bayesian ensemble of GBDT models is used to measure uncertainty by leveraging an algorithm called stochastic gradient Langevin boosting (SGLB), which combines GB with stochastic gradient MCMC (SG-MCMC). Although theoretically sound, SGLB gets trapped easily on a particular mode of the Bayesian posterior, just like other forms of SG-MCMCs. Therefore, a *single* SGLB model can often fail to produce uncertainty estimates of high-fidelity. To address this problem, we present Cyclical SGLB (cSGLB) which incorporates a Cyclical Gradient schedule in the SGLB algorithm. The cyclical gradient mechanism promotes new mode discovery and helps explore high multimodal posterior distributions. As a result, cSGLB can *efficiently* quantify uncertainty in GB with only a *single* model. In addition, we present another cSGLB variant with data bootstrapping to further encourage diversity among posterior samples. We conduct extensive experiments to demonstrate the efficiency and effectiveness of our algorithm, and show that it outperforms the state-of-the-art SGLB on uncertainty quantification, especially when uncertainty is used for detecting out-of-domain (OOD) data or distributional shifts.

Keywords

uncertainty quantification, gradient boosting decision trees, Bayesian inference, out-of-domain (OOD) detection

1. Introduction

With the rapid growth of data and computing power, machine learning (ML) has been gaining a lot of new applications in areas not imagined before. The more ubiquitous ML systems become, it is inevitable to see applications in very sensitive and high-risk fields. This expands to a numerous areas like criminal recidivism [1], medical follow-ups [2] and autonomous-systems [3]. While these systems might be very broad, they share a common need, and that is to have a certain degree of confidence on ML predictions. A proven successful way to build confidence in critical systems is uncertainty estimation. Research has shown that humans are more likely to agree with a system if they get access to the corresponding uncertainty, and this holds true regardless of shape and variance as the approach itself is model and task agnostic [4]. Since the most common data type in real-world ML applications is tabular [5], our work in this paper focuses specifically on uncertainty quantification for the state-of-the-art gradient boosting decision trees (GBDTs) [6, 7], which are known to outperform deep learning (DL) methods on tabular data, both in accuracy and tuning requirements [5]. Measuring uncertainty effectively and efficiently on GBDT predictions can therefore not only improve model interpretability in production but also ensure a safer deployment of ML systems, especially for high-risk applications.

Uncertainty quantification (UQ) has been widely studied for neural networks under the Bayesian framework [8], however, it is relatively under-explored for tree-based models. Although calibrated probability estimation trees [9, 10] can be used for UQ, they have not been studied from a Bayesian perspective. Recently, Bayesian ensemble methods were extended to measure uncertainty in GBDTs by leveraging a new algorithm called stochastic gradient Langevin boosting (SGLB) [11]. Specifically, two SGLB-based approaches were introduced for UQ [12]: (1) *SGLB ensemble*, which trains multiple SGLB models in parallel, and (2) *SGLB virtual ensemble*, which constructs a *virtual* ensemble using only a *single* SGLB model where each member in the ensemble is a "truncated" sub-model [12]. Although both approaches are theoretically sound, there is clearly a trade-off between quality and efficiency in practice. SGLB (real) ensemble is believed to be accurate as it can characterize the Bayesian posterior well by running independent models in parallel. However, it is almost infeasible to deploy such an ensemble in real-world production due to its high computational and maintenance costs. SGLB virtual ensemble greatly improves the efficiency, however, it often gets stuck on a single mode of the Bayesian posterior and can produce downgraded uncertainty estimates [13, 14]. To better balance between quality and efficiency and to facilitate

The AAAI-23 Workshop on Artificial Intelligence Safety (SafeAI 2023), February 13 - 14, 2023, Washington, D.C., USA

*Corresponding author.

tianta@amazon.com (T. Tan); carlohue@amazon.com (C. Huertas); qqzhao@amazon.com (Q. Zhao)

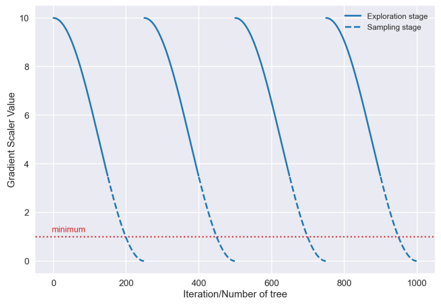


Figure 1: Illustration of proposed cyclical schedule on gradient scales for SGLB algorithm.

the usage of uncertainty-enabled ML systems, an important question remains: *how can we make a single SGLB explore effectively different modes of a posterior given a limited computational budget?*

In this paper, we address the question above by combining SGLB virtual ensemble with advanced sampling techniques from Bayesian DL [14, 13, 15, 16]. Inspired by the ideas in [13], we propose to use a scaler (or scaling factor) on gradients that follows a cyclical schedule during the course of SGLB training. The cyclical schedule is illustrated in Figure 1, and consequently, we name the resulting algorithm *Cyclical SGLB (cSGLB)*. Similar to [13], each cycle in cSGLB contains two stages: (1) *Exploration*: when the scaler is large, we treat this stage as a warm restart from the previous cycle, enabling the model/sampler to follow the gradients closely and to escape from the current local mode. (2) *Sampling*: when the gradient scaler is small, the scale of injected Gaussian noise in the SGLB procedure becomes relatively large, encouraging the sampler to fully characterize one local mode. We collect one sample (or truncated sub-model) to build the virtual ensemble at the end of each cycle. The cyclical gradient schedule therefore helps cSGLB *effectively* explore different modes of a posterior while maintaining the same level of *efficiency* of a virtual ensemble. Moreover, inspired by a recent study [16] showing that "diversified" posterior may provide a tighter generalization bound, we present another simple approach to encourage *diversity* in samples obtained from running cSGLB via *data bootstrapping*. We name this variant *Cyclical Bootstrapped SGLB (cbSGLB)*.

We extensively experiment with our proposed algorithms and compare the performance against SGLB ensemble and the original SGLB virtual ensemble. Particularly, we show that our cyclical gradient schedule can help explore effectively multimodal distributions, cSGLB is capable of producing uncertainty estimates that are better aligned with SGLB real ensemble, and cSGLB/cbSGLB outperforms the SGLB baseline with a large margin on out-of-domain (OOD) data detection, indicating a superior performance in detecting distributional/domain shifts in real-world tabular data streams.

2. Related Work

Bayesian ML and approximate Bayesian inference provide a principled representation of uncertainty. One popular family of approaches to inference in Bayesian ML are stochastic gradient Markov Chain Monte Carlo (SG-MCMC) methods [17, 18, 19, 20, 13], which are used to effectively sample models (or model parameters) from the Bayesian posterior. Uncertainty then comes naturally by measuring the "discrepancy" in predictions from the sampled models which are regarded as *posterior samples*. Recently, stochastic gradient Langevin boosting (SGLB) [11] was proposed by combining gradient boosting with SG-MCMC. As its name suggests, the Markov chain generated by SGLB obeys a special form of the stochastic gradient Langevin dynamics (SGLD) [11, 17], which implies that SGLB is able to generate samples from the true Bayesian posterior asymptotically. Leveraging this property, Malinin et al. [12] proposed to use (1) SGLB ensemble or (2) SGLB virtual ensemble to measure uncertainty in GBDTs. Essentially, *SGLB ensemble* corresponds to running multiple SG-MCMCs in parallel and each chain (or SGLB) is initialized independently with a different random seed. Since SGLB allows us to sample from the true posterior, the ensemble with multiple samples gives a high-fidelity approximation to the Bayesian posterior. In contrast, *SGLB virtual ensemble* only trains a single SGLB model and it uses multiple truncated sub-models to form a (virtual) ensemble. The key idea is essentially extracting multiple samples from a single-chain SG-MCMC instead of running multiple chains in parallel.

In theory, SGLB or single-chain SG-MCMC converges asymptotically to the target distribution and should behave similarly to the multi-chain SGLB ensemble in the limit, but it can suffer from a bounded estimation error in limited time [21]. Moreover, it is often believed that the posterior is highly multimodal in the parametric space of modern ML models [13], since there are potentially many different sets of parameters that can describe the training data equally well. The real ensemble can explore different modes of the posterior by running in parallel independent chains, providing a complete picture of the distribution as the number of chains increases. However, a single-chain SG-MCMC often gets stuck easily on a single mode of the posterior [13, 14], failing to cover the full spectrum of the distribution.

In this paper, we extend the ideas behind Cyclical SG-MCMC (cSG-MCMC) in DL [13] to sampling from a tree-based SGLB model, which promotes new mode discovery during training. Different from cSG-MCMC that puts a cyclical schedule on step size, we propose to use a cyclical schedule on gradient scale. We also point out and justify the difference and our design choice in Appendix A. In addition, we propose a simple strategy to further encourage diversity in samples obtained from a single chain by data bootstrapping. At the beginning of each cycle (see Fig.1), we construct a bootstrapped dataset that is a random subset of the training, and use that bootstrapped data *consistently* during the exploration stage to update the GBDT model. The "bias" induced by data bootstrapping also amounts to posterior tempering [14, 15, 22, 23, 13].

3. Preliminaries

3.1. General Setup

Given a set of N training data points sampled from an unknown distribution $\mathcal{D}$ on $\mathcal{X} \times \mathcal{Y}$, i.e., $(x_1, y_1), \ldots, (x_N, y_N) \sim \mathcal{D}$ denoted as $\mathcal{D}_N$, and a loss function $L(z, y) : \mathcal{Z} \times \mathcal{Y} \to \mathbb{R}$ where $\mathcal{Z}$ denotes the space of predictions, our goal is to minimize the empirical loss $\mathcal{L}(f|\mathcal{D}_N) := \frac{1}{N}\sum_{i=1}^{N} L(f(x_i), y_i))$ over functions f belonging to some family $\mathcal{F} \subset \{f : \mathcal{X} \to \mathcal{Z}\}$. In this paper, we only consider $\mathcal{F}$ corresponding to additive ensemble of decision trees $\mathcal{H} := \{h^s(x, \theta^s) : \mathcal{X} \times \mathbb{R}^{m_s} \to \mathbb{R}, s \in S\}$, where S is an index set and h^s has parameters θ^s. Decision trees are built by partitioning recursively the feature space into disjoint regions (called leaves). Each region is assigned a value that is used to estimate the response of y in the corresponding feature subspace. Let's denote these regions by R_j's, then we have $h(x, \theta) = \sum_j \theta_j \mathbb{1}\{x \in R_j\}$, where $\mathbb{1}\{\cdot\}$ denotes indicator function. Therefore, given the tree structure, decision tree h^s is a linear function of its parameters θ^s. It is often assumed that the set S is *finite* because the training data is finite [11, 12], e.g., there exists only a finite number of ways to partition the training data. Owing to the linear dependence of h^s on θ^s and the finite assumption of S, we can represent any ensemble of models from $\mathcal{H}$ as a linear model $f_\Theta(x) = \phi(x)^T \Theta$ for some feature map $\phi(x) : \mathcal{X} \to \mathbb{R}^m$ and $\Theta \in \mathbb{R}^m$ denotes the parameters of the entire ensemble [11]. Hence, in the subsequent discussion, we will simply denote the parameters of the GBDT model obtained at iteration τ as $\hat{\Theta}_\tau$, and additionally define a linear mapping $H_s : \mathbb{R}^{m_s} \to \mathbb{R}^N$ that converts θ^s to predictions $(h^s(x_i, \theta^s))_{i=1}^N$.

3.2. SGLB

SGLB combines stochastic gradient boosting (SGB) [7] with stochastic gradient Langevin dynamics (SGLD) [17]. Following notations used in the original paper [11], we characterize the SGB procedure by a tuple $\mathcal{B} := \{\mathcal{H}, p\}$, where $\mathcal{H}$ again is the set of base learners and $p(s|g)$ is a distribution over indices $s \in S$ conditioned on a gradient vector $g \in \mathbb{R}^N$. Simply put, $p(s|g)$ defines a distribution over tree structures.

As with other GBDT algorithms, SGLB constructs an ensemble of decision trees iteratively. At each iteration τ, we compute unbiased gradient estimates $\hat{g}_\tau$ such that $\mathbb{E}[\hat{g}_\tau] = (\frac{\partial}{\partial f} L(f_{\hat{\Theta}_\tau}(x_i), y_i))_{i=1}^N \in \mathbb{R}^N$ using the current model $f_{\hat{\Theta}_\tau}$, and sample independently two normal vectors $\zeta, \zeta' \sim \mathcal{N}(0_N, I_N)$, where $0_N, I_N$ denote zero vector and identity matrix in $\mathbb{R}^N$, respectively. Then, a base learner (or tree structure) s_τ is picked by drawing one sample from $p(s|\hat{g}_\tau + \sqrt{\frac{2N}{\epsilon\beta}}\zeta')$, where $\epsilon > 0$ is a learning rate (or step size) and $\beta > 0$ is a parameter often referred as inverse diffusion temperature. Next, we estimate the parameters $\theta_*^{s_\tau}$ (at tree leaves) of the sampled base learner by solving the following optimization:

$$\begin{aligned} &\text{minimize } ||\theta^{s_\tau}||_2^2 \quad s.t. \\ &\theta^{s_\tau} \in \underset{\theta \in \mathbb{R}^{m_{s_\tau}}}{\operatorname{argmin}} || - \hat{g}_\tau + \sqrt{\frac{2N}{\epsilon\beta}}\zeta - H_{s_\tau}\theta||_2^2, \end{aligned} \tag{1}$$

which returns the minimum norm solution that fits best to the perturbed "noisy" version of negative gradients. The optimization above has a closed form solution as $\theta_*^{s_\tau} = -\Phi_{s_\tau}(\hat{g}_\tau + \sqrt{\frac{2N}{\epsilon\beta}}\zeta)$, where $\Phi_{s_\tau} := (H_{s_\tau}^T H_{s_\tau})^+ H_{s_\tau}^T$, $^+$ denotes pseudo-inverse. For decision trees, $\Phi_{s_\tau} g$ essentially corresponds to averaging the gradient estimates g in each leaf node of the tree. Lastly, SGLB algorithm updates the ensemble model by

$$f_{\hat{\Theta}_{\tau+1}}(\cdot) := (1 - \gamma\epsilon) f_{\hat{\Theta}_\tau}(\cdot) + \epsilon h^{s_\tau}(\cdot, \theta_*^{s_\tau}), \tag{2}$$

where γ is a regularization parameter that "shrinks" the currently built model when updating the ensemble. At a high-level, SGLB is a stochastic GB algorithm with Gaussian noise injected into gradient estimates, which encourages the algorithm to explore a larger area in the functional space to find a better fit for the given data. The independence between noise ζ (used for parameter learning) and ζ' (used for tree sampling), and the model shrinking by γ in Eqn.(2) are technical details needed for establishing theoretical results and rigorous analysis of SGLB [11]. All the procedures of SGLB are also present in our proposed cSGLB in Algo. 1 (with our additional modifications highlighted in blue).

One can show that the parameters of SGLB $\hat{\Theta}_\tau$ at each iteration form a Markov chain that weakly converges to the following stationary distribution:

$$p_*^\beta(\Theta) \propto \exp(-\beta\mathcal{L}(\Theta|\mathcal{D}_N) - \beta\gamma||\Gamma\Theta||_2^2), \tag{3}$$

where $\Gamma = \Gamma^T > 0$ is a regularization matrix which depends on a particular tree construction algorithm or the choice of tuple $\mathcal{B} := \{\mathcal{H}, p(s|g)\}$ [11]. Note that since the GBDT model is linear and can be fully determined by parameters Θ, we simply use notation $\mathcal{L}(f|\mathcal{D}_N)$ and $\mathcal{L}(\Theta|\mathcal{D}_N)$ interchangeably.

3.3. Posterior Sampling

We consider here a standard Bayesian learning framework [8] that treats parameters Θ as random variables and places a prior $p(\Theta)$ over Θ. In addition, we consider the GBDT model f_Θ as a *probabilistic model* and explicitly denote the model by $P(y|x;\Theta)$ with parameters Θ. This is valid naturally for classification as GBDT models by construction return a distribution over class labels. For regression, one can leverage NGBoost algorithm [24] to return the mean and variance of a Gaussian distribution over the target y for a given input x.

For the purpose of uncertainty estimation, we aim to estimate or obtain an approximation to the Bayesian posterior $p(\Theta|\mathcal{D}_N)$. To that end, we can choose $\beta = N$ and $\gamma = \frac{1}{2N}$ and use the negative log-likelihood as the loss function $\mathcal{L}(\Theta|\mathcal{D}_N) = \mathbb{E}_{\mathcal{D}_N}[-\log p(y|x,\Theta)] = -\frac{1}{N}\sum_{i=1}^{N}\log p(y_i|x_i,\Theta)$. Then, the limiting distribution of SGLB can be explicitly expressed as:

$$\begin{aligned} p_*^\beta(\Theta) &\propto \exp\Big(\log p(\mathcal{D}_N|\Theta) - \frac{1}{2}||\Gamma\Theta||_2^2\Big) \\ &\propto p(\mathcal{D}_N|\Theta)p(\Theta), \end{aligned} \tag{4}$$

which is proportional to the true posterior $p(\Theta|\mathcal{D}_N)$ under Gaussian prior $p(\Theta) = \mathcal{N}(0_m, \Gamma)$ [11].

Now, consider a Bayesian ensemble of probabilistic models $\{P(y|x;\Theta^{(k)})\}_{k=1}^K$ where each model is trained independently by running SGLB. Since each $\Theta^{(k)}$ is guaranteed to be sampled from $p(\Theta|\mathcal{D}_N)$ by Eqn.(4), the ensemble $\{\Theta^{(k)}\}_{k=1}^K$ with K samples yields a "discrete" approximation to the posterior $p(\Theta|\mathcal{D}_N)$. This is exactly the idea behind *SGLB ensemble* [12], which learns K independent SGLB models in parallel with different random seeds. Although the approximation improves as K increases, the computational cost also increases linearly with K. To alleviate the computational burden, *SGLB virtual ensemble* [12] builds a Bayesian *virtual* ensemble by sampling multiple times from a single-chain SGLB model. Because samples from the same chain are *highly correlated*, SGLB virtual ensemble proposes to sample one member $\Theta^{(k)}$ every $C > 1$ iterations. More specifically, the parameters are sampled by $\{\Theta^{(k)}\}_{k=1}^{K=\lfloor\frac{\mathcal{T}}{2C}\rfloor} = \{\hat{\Theta}_{C(k+\lfloor\frac{\mathcal{T}}{2C}\rfloor)}, k = 1,\ldots,\lfloor\frac{\mathcal{T}}{2C}\rfloor\}$, i.e., appending one member to the ensemble every C iterations while constructing one SGLB model using $\mathcal{T}$ iterations of gradient boosting. Notice that no sampling is performed during the first half of iterations ($\tau < \mathcal{T}/2$) since Eqn.(4) holds only asymptotically. For large C and K, the virtual ensemble should behave similarly to the SGLB real ensemble in the limit theoretically.

3.4. Uncertainty Estimation

Once the Bayesian (virtual) ensemble $\{P(y|x;\Theta^{(k)})\}_{k=1}^K, \Theta^{(k)} \sim p(\Theta|\mathcal{D}_N)$ is learned, predictions can be made by taking an average over the ensemble, often known as *predictive posterior* or *Bayesian model average (BMA)*:

$$p(y|x,\mathcal{D}_N) = \mathbb{E}_{p(\Theta|\mathcal{D}_N)}[P(y|x;\Theta)] \approx \tfrac{1}{K}\textstyle\sum_{k=1}^K P(y|x;\Theta^{(k)}). \tag{5}$$

The entropy of the predictive posterior estimates *total uncertainty (TU)* in predictions, which can be further decomposed into two distinct types of uncertainty: *knowledge uncertainty* and *data uncertainty.*[1] (a) Knowledge uncertainty (KU) arises due to the *lack of knowledge* about the data generation process (or the unknown distribution $\mathcal{D}$). KU is expected to be large in regions (in the feature space) where we do not have sufficient training data. (b) Data uncertainty (DU) arises due to the *inherent stochasticity* within the data generation process, and it is high in regions with class overlaps. In applications like active learning [25], reinforcement learning (RL) [26], and OOD detection, it is desirable to measure KU separately from DU (or TU), and the following equation can be used in practice to compute and connect them via *mutual information* [27]:

$$\begin{aligned} &\underbrace{\mathbb{I}(y;\Theta|x,\mathcal{D}_N)}_{\text{Knowledge Uncertainty}} \\ &= \underbrace{\mathbb{H}(p(y|x,\mathcal{D}_N))}_{\text{Total Uncertainty}} - \underbrace{\mathbb{E}_{p(\Theta|\mathcal{D}_N)}[\mathbb{H}(P(y|x;\Theta))]}_{\text{Expected Data Uncertainty}} \\ &\approx \mathbb{H}\Big(\frac{1}{K}\sum_{k=1}^K P(y|x;\Theta^{(k)})\Big) - \frac{1}{K}\sum_{k=1}^K \mathbb{H}\Big(P(y|x;\Theta^{(k)})\Big), \end{aligned} \tag{6}$$

where $\mathbb{I}(A;B)$ denotes the mutual information between random variables A and B, and $\mathbb{H}(\cdot)$ denotes entropy. The difference between TU and DU measures the disagreement among members in the ensemble and estimates the knowledge uncertainty.[2]

4. Cyclical Stochastic Gradient Langevin Boosting (cSGLB)

4.1. Promoting Mode Discovery via Cyclical Gradient Scheduling

Instead of building a cumbersome true ensemble of SGLB models, the virtual ensemble of SGLB greatly improves the efficiency by training only a single model. However, similar to other types of SG-MCMC in Bayesian DL [13, 14, 15], single-chain SGLB gets trapped easily on a particular single mode of the posterior. To efficiently explore different modes of the multimodal posterior and effectively measure uncertainty in GBDT predictions with

[1]KU is also named epistemic uncertainty and DU is also called aleatoric uncertainty.
[2]See paper [12] for equations computing KU and DU in regression tasks.

a single chain, we propose a simple remedy that places a cyclical cosine schedule on gradient scale during training, as illustrated in Fig.1. Specifically, the scaling factor at iteration τ is defined as:

$$\alpha_\tau = \max\left(\frac{\alpha_{\max}}{2}\left[\cos(\frac{\pi \mod(\tau, C)}{C})+1\right],\ \alpha_{\min}\right), \tag{7}$$

where $\alpha_{\max} \geq 1$ is the maximum of the scaler or the initial value of α_0, C is the user-defined cycle length, and $\alpha_{\min}$ defines the minimum of the scaler, e.g., $\alpha_{\min} = 1$ or 0.5, since decaying the gradients to arbitrarily small could be harmful for performance. Putting together, this amounts to sampling the tree structure and learning the tree leaf parameters with the (re)scaled gradients: $s_\tau \sim p(s|\alpha_\tau \hat{g}_\tau + \sqrt{\frac{2N}{\epsilon\beta}}\zeta')$ and $\theta_*^{s_\tau} = -\Phi_{s_\tau}(\alpha_\tau \hat{g}_\tau + \sqrt{\frac{2N}{\epsilon\beta}}\zeta)$.

Similar to Cyclical SG-MCMC [13], we define two stages within each cycle: (1) *Exploration* when the completed portion of a cycle $\mathcal{A}(\tau) = \frac{\mod(\tau, C)}{C}$ is smaller than a given threshold: $\mathcal{A}(\tau) < \eta$, and (2) *Sampling* when $\mathcal{A}(\tau) \geq \eta$, and $\eta \in (0, 1)$ balances the portion between exploration and sampling. We obtain one sample from the chain at the end of each cycle, i.e., the virtual ensemble is built by $\{\Theta^{(k)}\}_{k=1}^{K=\lfloor\frac{T}{C}\rfloor} = \{\hat{\Theta}_{Ck-1}, k = 1, \ldots, \lfloor\frac{T}{C}\rfloor\}$. Large gradients at the beginning of a cycle provide enough perturbation and encourage the model to escape from the current mode, while decreasing the gradient scale inside one cycle makes the sampler better characterize the density of the local mode. Moreover, many prior works in Bayesian NNs proposed to apply a certain form of *preconditioning* to compensate sampling noises from mini-batch training [15, 14]. Tree-based models can usually digest the full-batch (full dataset $\mathcal{D}_N$) per iteration by leveraging modern multi-core processors and multi-threading. Therefore, we directly use the full-batch GB in all *sampling* stages, while leaving the option of random data subsampling in *exploration* stages to the users if training time is a concern.

Combining cyclical gradient scaling with SGLB, we expect that our new Cyclical SGLB (cSGLB) algorithm could inherit most (if not all) theoretical properties of the original SGLB algorithm. Conceptually, with a proper choice of $\alpha_{\max}, \alpha_{\min}$ and cycle length C, the sample obtained at $\tau = C - 1$ from the Markov chain $f_{\hat{\Theta}_\tau}$ generated by Algo. 1 (w/o bootstrap) can be approximately seen as a random draw from the limiting distribution with small bounded errors. Also, each next cycle can be viewed as a warm restart from its previous cycle, and thus no errors shall be accumulated into the subsequent cycles (at sampling time $\tau = Ck - 1$). We left rigorous analysis and proofs of our propositions for future work. Empirically, we show in our experiments that the cyclical gradient scaling achieves similar effects in exploring a multimodal distribution when compared with cSG-MCMC which places a similar cyclical schedule on step size within the context of Bayesian DL. In fact, cSGLB extends the idea behind cSG-MCMC to tree-based GBDT models. We also summarize key differences between our design of cSGLB and cSG-MCMC in Appendix A.

4.2. Enhancing Sample Diversity via Bootstrapping

Recent work [16] provided a compelling analysis that the Bayesian posterior is *not* optimal under model misspecification[3], where the performance of the true posterior is dominated by an alternative non-Bayesian posterior that explicitly encourages *diversity* among ensemble member predictions. Inspired by these results, we propose a simple strategy that promotes diversity among samples obtained from cSGLB by *data bootstrapping*. At the beginning of each cycle, we sample randomly a *Bernoulli mask* of size N, i.e., $\nu := \{\nu[i] \sim Bernoulli(p_{bm})\}_{i=1}^N \in \{0, 1\}^N$, and $p_{bm} \in (0, 1)$ defines the percentage of data being used. In the following *exploration* stage, we mask out gradients $\hat{g}_\tau$ by taking an *element-wise* product with ν, i.e., $\hat{g}_\tau \odot \nu$. The mask ν and the mask-out operation are used *consistently* throughout the exploration stage $\mathcal{A}(\tau) < \eta$, and ν only gets updated until the end of the cycle. This design amounts to learning with a bootstrapped subsample of the data in each cycle. Since the model would observe consistently less data than the original $\mathcal{D}_N$, it also amounts to posterior tempering $(p(\mathcal{D}_N|\Theta)p(\Theta))^{1/T}$ with some temperature $T > 1$, resulting in a *warm posterior* that is softer than the Bayesian posterior. By increasing the temperature T, we expect to see increased density on the paths/corridors connecting different modes of the posterior [28, 29], further facilitating the sampler to escape from the current local mode. By using a relatively large $\eta \in (0.8, 1)$, the tempering effects would carry over into the sampling stage. Therefore, the bootstrapping mechanism helps improve the sample diversity from cSGLB, and we name this variant *Cyclical Bootstrapped SGLB (cbSGLB)*.

Lastly, we summarize our proposed cSGLB (plus bootstrap option) in Algo. 1 and highlight our modifications on top of SGLB in blue.

5. Experiments

5.1. Experiments on Synthetic Data

We validate and qualitatively evaluate the proposed gradient scheduling and our cSGLB algorithm on two synthetic problems: (1) a synthetic multimodal dataset in [13], and

[3]The function class in-use does not contain the unknown ground-truth function.

```
Algorithm 1: Cyclical (Bootstrapped) SGLB
  Input: dataset $\mathcal{D}_N$, learning rate $\epsilon > 0$, inverse
    temperature $\beta > 0$, regularization $\gamma > 0$,
    number of iterations $\mathcal{T} > 0$, cycle length $C > 1$,
    scaler limits $\alpha_{\max}, \alpha_{\min} > 0$, stage threshold
    $\eta > 0$, mask probability $p_{bm} > 0$, boolean
    indicator bootstrap

  Initialize $f_{\hat{\Theta}_0}(\cdot) = 0$, $\nu = 1_N \in \mathbb{R}^N$ as all-ones
    vector
  for $\tau$ in $[0, 1, \ldots, \mathcal{T} - 1]$ do
      if bootstrap then
          if $\frac{\mathrm{mod}\ (\tau, C)}{C} = 0$ then
              Sample $\nu \in \mathbb{R}^N$ with
                $\nu[i] \sim Bernoulli(p_{bm})$
          end
          if $\frac{\mathrm{mod}\ (\tau, C)}{C} \geq \eta$ then
              Set $\nu = 1_N$
          end
      end
      Compute gradient scaler: $\alpha_\tau =$
        $\max(\frac{\alpha_{\max}}{2}[\cos(\frac{\pi \ \mathrm{mod}\ (\tau, C)}{C}) + 1], \ \alpha_{\min})$
      Estimate gradients $\hat{g}_\tau$ using $f_{\hat{\Theta}_\tau}(\cdot)$ and $\mathcal{D}_N$:
        $\hat{g}_\tau = (\frac{\partial}{\partial f} L(f_{\hat{\Theta}_\tau}(x_i), y_i))_{i=1}^N \in \mathbb{R}^N$
      Sample noise $\zeta, \zeta' \sim \mathcal{N}(0_N, I_N)$
      Sample tree structure:
        $s_\tau \sim p(s | \alpha_\tau (\hat{g}_\tau \odot \nu) + \sqrt{\frac{2N}{\epsilon\beta}} \zeta')$
      Estimate leaf/parameter values:
        $\theta_*^{s_\tau} = -\Phi_{s_\tau} (\alpha_\tau (\hat{g}_\tau \odot \nu) + \sqrt{\frac{2N}{\epsilon\beta}} \zeta)$
      Update GBDT model:
        $f_{\hat{\Theta}_{\tau+1}}(\cdot) = (1 - \gamma\epsilon) f_{\hat{\Theta}_\tau}(\cdot) + \epsilon h^{s_\tau}(\cdot, \theta_*^{s_\tau})$
  end
  Return: $f_{\hat{\Theta}_\mathcal{T}}(\cdot)$
```

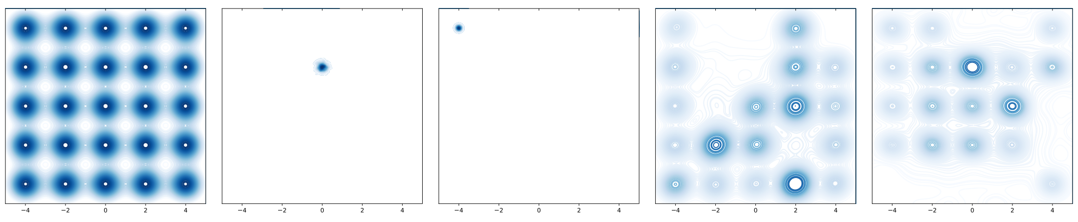
(2) a multi-class Spiral dataset in [12]. Due to limited space, we include experimental details in Appendix B.

Synthetic Multimodal Data We first demonstrate the ability of cyclical gradient scaling for sampling from a multimodal distribution on a 2D mixture of 25 Gaussians. Specifically, we compare (i) the original SG-MCMC with SGLD (denoted as SGLD) and two SGLD variants: (ii) SGLD with Cyclical schedule on Learning Rate (denoted as clr-SGLD) [13] and (iii) SGLD with Cyclical schedule on Gradient scale (denoted as cg-SGLD (*ours*)). We reproduced the results for SGLD and clr-SGLD in paper [13] with code released by the authors, and built our cg-SGLD on top of it. In addition, we experimented with a "noisy" version of SGLD with a fixed lr and increased $10\times$ noise scale (denoted as NoisySGLD/N-SGLD). For a fair comparison, each chain runs for $50k$ iterations and both clr-SGLD and cg-SGLD have 30 cycles. Fig.2 shows the estimated density using different sampling strategies. SGLD gets trapped in local modes depending on the random initial position, and increasing the noise scale does not solve the problem. In contrast, clr-SGLD and cg-SGLD can explore and locate roughly $7-8$ different modes of the distribution, showing that our cg-SGLD can achieve the state-of-the-art performance in exploring multimodal distributions. Moreover, cg-SGLD has benefits in implementation over clr-SGLD when combined with SGLB. The SGLB algorithm was made available in the CatBoost library [30], which only supports a fixed lr. All our proposed enhancements can be implemented with a "user-defined loss function" available in CatBoost *without* touching the source code, making it straightforward to reproduce our algorithms.

Multi-Class Spiral Data After validating the efficacy of cyclical gradient scheduling on sampling from multimodal distributions, we are now ready to experiment with cSGLB. Specifically, we compare the following algorithms on a 3-class classification task called "Spiral" in [12]: (i) SGLB ensemble, where we denote by *ens*$_K$ with K models, (ii) SGLB virtual ensemble, simply denoted by *SGLB*, and (iii) cSGLB virtual ensemble, denoted by *cSGLB* (ours). We again reproduced the results in [12] with code released by the authors, and Fig.3 shows the estimated KU on Spiral test set. As noted in [12], we see that knowledge uncertainty due to decision-boundary "jitter" exists in both *ens*$_{20}$ and *cSGLB*, and the "jitter" affects *cSGLB* more as the estimated KU is "noisy" at the decision boundary. Nevertheless, *cSGLB* (with only a *single* model) is significantly more efficient than *ens*$_{20}$ and is able to greatly improve upon *SGLB* in capturing high KU in regions with no training data.

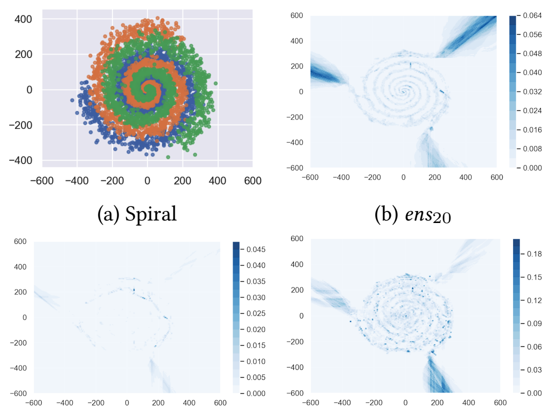
5.2. Experiments on Real-World Weather Prediction Data

Lastly, we evaluate our proposed methods on the public Shifts Weather Prediction dataset [31]. We select the classification task where the ML model is asked to predict the precipitation class at a particular geolocation and timestamp, given heterogeneous tabular features derived from weather station measurements and forecast models. The full dataset is partitioned in a canonical fashion and contains in-domain (ID) training, development and evaluation datasets as well as out-of-domain (OOD) development and evaluation datasets. Importantly, the ID data and the OOD data are separated in time and consist of non-overlapping climate types (ID: *Tropical, Dry, Mild*; OOD: *Snow, Polar*), making the Shifts dataset an ideal testbed for gauging the robustness of ML model and the quality of uncertainty estimation. To further facilitate our experimentation, we conducted the following



(a) Target (b) SGLD (c) N-SGLD (d) clr-SGLD (e) cg-SGLD (*ours*)

Figure 2: Sampling from a 5 by 5 mixture of 25 Gaussians.



(a) Spiral (b) *ens*$_{20}$ (c) *SGLB* (d) *cSGLB*

Figure 3: Spiral dataset and estimated knowledge uncertainties. Each different color in (a) represents a different class.

data preprocessing: (1) feature selection to keep only the top 40 features by importance (out of 123 available features), where the feature importance is determined by a CatBoost classifier with $1K$ trees trained on the entire training set; (2) dropping minority classes to keep only the major 3 precipitation classes, i.e., class 0, 10, and 20 out of the 9 available classes from the original dataset; (3) random data sampling to keep $200K$ (medium-sized) data in the final training set. Again, the purpose of our data preprocessing is for speeding up experimentation, and we believe that the observations and findings in this study are generalizable to the original full dataset. For model building, 30 independent SGLB models (each of $1K$ trees) were trained and used to construct real ensembles ens_K for $K \in \{3, 5, 10, 30\}$. SGLB/cSGLB/cbSGLB virtual ensembles were built by sampling 10 members from a single-chain with $2K$ trees. Hence, ens_{10} is $5\times$ more expensive in computation and memory than a virtual ensemble. Additional details regarding our data and models are included in Appendix C.

We compare various methods on their predictive performance and on uncertainty quantification following [31], and the results are summarized in Table 1. For predictive performance, we report the classification accuracy and macro F1 using BMA on both the ID and the OOD evaluation datasets. We can see the following effects: (1) Virtual ensembles with a longer chain slightly outperform the real ensembles on the ID data. (2) Our proposed cSGLB and cbSGLB perform slightly worse than the rest of methods on the OOD data. However, this usually is not a concern in practice since the model is *not* trained with data from OOD domains and would not be used to solve the OOD prediction tasks in a practical scenario. As long as the domain shifts can be reliably detected (via uncertainty), proactive decisions can be made to avoid costly mistakes due to model errors. (3) Our proposed data bootstrapping mechanism is capable of improving the performance on the OOD data (cbSGLB $>$ cSGLB). In addition, we include the F1-AUC metric (on the combined ID&OOD evaluation sets) introduced in [31] to jointly assess the predictive power and uncertainty quality. The F1-AUC can be increased by either having a stronger predictive model or by improving the correlation between uncertainty and error. Consistent with the findings in [31], total uncertainty (TU) correlates more with errors than knowledge uncertainty (KU) as shown by the F1-AUC scores. More specifically, we see that the F1-AUC is quite similar across the board when measured by TU, although cSGLB/cbSGLB has slightly worse predictive power on the OOD segment. When F1-AUC is measured by KU, our cSGLB/cbSGLB is capable of producing KU estimates that relate more closely to model errors than KU from the SGLB baseline.

At last, we present the OOD detection ROC-AUC performance on the evaluation data by using KU estimates. Our cSGLB/cbSGLB outperforms the SGLB baseline with a large margin on the OOD detection task, and even achieves a comparable performance to the real ensemble ens_{10}, which is $5\times$ more expensive. This highlights that our cSGLB/cbSGLB can produce high-fidelity KU estimates to detect domain (or distributional) shifts with a single model, and that our proposed cyclical gradient scheduling is effective in exploring different modes of a posterior. In real-world industrial applications, detecting

Metric	Data	ens_3	ens_5	ens_{10}	ens_{30}	*SGLB*	*cSGLB*	*cbSGLB*
Accuracy (%)↑	*eval-ID*	65.24 ± 0.02	65.25 ± 0.02	65.26 ± 0.01	65.26	65.63 ± 0.01	65.93 ± 0.01	65.59 ± 0.01
	eval-OOD	52.51 ± 0.06	52.55 ± 0.06	52.55 ± 0.03	52.54	**52.50** ± 0.14	50.72 ± 0.17	51.49 ± 0.11
Macro F1 (%)↑	*eval-ID*	63.28 ± 0.02	63.29 ± 0.02	63.30 ± 0.01	63.30	63.69 ± 0.01	64.06 ± 0.02	63.67 ± 0.01
	eval-OOD	52.36 ± 0.07	52.39 ± 0.06	52.38 ± 0.03	52.38	**52.36** ± 0.13	50.64 ± 0.16	51.41 ± 0.15
F1-AUC (%)↑	*TU*	57.18 ± 0.01	57.19 ± 0.01	57.20 ± 0.01	57.20	57.26 ± 0.01	56.82 ± 0.03	56.95 ± 0.04
	KU	52.94 ± 0.03	53.71 ± 0.03	54.33 ± 0.02	54.83	52.55 ± 0.08	**53.60** ± 0.08	**53.31** ± 0.14
OOD-AUC (%)↑	*KU*	65.72 ± 0.31	68.60 ± 0.21	71.15 ± 0.16	73.26	67.32 ± 0.70	**71.45** ± 0.67	**71.70** ± 0.91

Table 1
Comparing predictive performance & uncertainty measures of various methods on Shifts Weather data. Mean ± Std Dev over 3 seeds. The best performance among virtual ensembles is highlighted in **bold**.

OOD data or domain shifts in an efficient way is often crucial to ensure a safe deployment and operation of ML systems. Observing consistently high uncertainty (especially KU) from model predictions indicates that the patterns of new incoming data have deviated from the training. This often provides a strong signal for model refresh, ensuring that the ML system can be updated in time to avoid errors and operate safely in its "comfort zone" (with relatively low predictive uncertainty).

6. Conclusion

We present cyclical gradient scheduling and Cyclical SGLB for *efficiently and effectively* quantifying uncertainty in gradient boosting with a *single* model, and propose a data bootstrapping scheme to enhance *diversity* in posterior samples. We show empirically that our algorithms have superior performance over the state-of-the-art SGLB, especially in quantifying knowledge uncertainty and for OOD detection.

Accurately quantifying uncertainty in ML predictions can yield many benefits in real-world applications. Uncertainty directly measures the confidence in model predictions, not only improving interpretability, but also ensuring that costly mistakes can be avoided proactively. Consistently observing high uncertainty in predictions on incoming data stream often provides a reliable signal for data distributional shifts and/or model being obsolete. Uncertainty is also a key concept in optimal control and decision making, where it can be leveraged to experiment with different actions/decisions with controllable costs in search for the best operating point of an ML system. We believe that our work on efficient uncertainty quantification for GBDTs facilitates the adoption of uncertainty-enabled and uncertainty-aware ML systems, and more broadly promotes ML safety and a safe and trustworthy deployment of ML model in production.

References

[1] J. Dressel, H. Farid, The accuracy, fairness, and limits of predicting recidivism, Science Advances 4 (2018) eaao5580. URL: https://www.science.org/doi/abs/10.1126/sciadv.aao5580. doi:10.1126/sciadv.aao5580.

[2] P. Stone, R. Brooks, E. Brynjolfsson, R. Calo, O. Etzioni, G. Hager, J. Hirschberg, S. Kalyanakrishnan, E. Kamar, S. Kraus, et al., Artificial intelligence and life in 2030, One Hundred Year Study on Artificial Intelligence: Report of the 2015-2016 Study Panel 52 (2016).

[3] A. Serban, E. Poll, J. Visser, Towards using probabilistic models to design software systems with inherent uncertainty (2020). URL: https://arxiv.org/abs/2008.03046. doi:10.48550/ARXIV.2008.03046.

[4] S. McGrath, P. Mehta, A. Zytek, I. Lage, H. Lakkaraju, When does uncertainty matter?: Understanding the impact of predictive uncertainty in ml assisted decision making (2020). URL: https://arxiv.org/abs/2011.06167. doi:10.48550/ARXIV.2011.06167.

[5] R. Shwartz-Ziv, A. Armon, Tabular data: Deep learning is not all you need, 2021. URL: https://arxiv.org/abs/2106.03253. doi:10.48550/ARXIV.2106.03253.

[6] J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of statistics (2001) 1189–1232.

[7] J. H. Friedman, Stochastic gradient boosting, Computational statistics & data analysis 38 (2002) 367–378.

[8] A. Malinin, Uncertainty estimation in deep learning with application to spoken language assessment, Ph.D. thesis, University of Cambridge, 2019.

[9] F. Provost, P. Domingos, Tree induction for probability-based ranking, Machine learning 52 (2003) 199–215.

[10] U. Johansson, T. Löfström, H. Boström, Calibrating probability estimation trees using venn-abers predictors, in: Proceedings of the 2019 SIAM International Conference on Data Mining, SIAM, 2019, pp. 28–36.

[11] A. Ustimenko, L. Prokhorenkova, Sglb: Stochastic gradient langevin boosting, in: International Conference on Machine Learning, PMLR, 2021, pp. 10487–10496.

[12] A. Malinin, L. Prokhorenkova, A. Ustimenko, Uncertainty in gradient boosting via ensembles, arXiv preprint arXiv:2006.10562 (2020).

[13] R. Zhang, C. Li, J. Zhang, C. Chen, A. G. Wilson, Cyclical stochastic gradient mcmc for bayesian deep learning,

arXiv preprint arXiv:1902.03932 (2019).

[14] F. Wenzel, K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, S. Nowozin, How good is the bayes posterior in deep neural networks really?, arXiv preprint arXiv:2002.02405 (2020).

[15] P. Izmailov, S. Vikram, M. D. Hoffman, A. G. G. Wilson, What are bayesian neural network posteriors really like?, in: International Conference on Machine Learning, PMLR, 2021, pp. 4629–4640.

[16] A. Masegosa, Learning under model misspecification: Applications to variational and ensemble methods, Advances in Neural Information Processing Systems 33 (2020) 5479–5491.

[17] M. Welling, Y. W. Teh, Bayesian learning via stochastic gradient langevin dynamics, in: Proceedings of the 28th international conference on machine learning (ICML-11), Citeseer, 2011, pp. 681–688.

[18] T. Chen, E. Fox, C. Guestrin, Stochastic gradient hamiltonian monte carlo, in: International conference on machine learning, PMLR, 2014, pp. 1683–1691.

[19] N. Ding, Y. Fang, R. Babbush, C. Chen, R. D. Skeel, H. Neven, Bayesian sampling using stochastic gradient thermostats, Advances in neural information processing systems 27 (2014).

[20] C. Li, C. Chen, D. Carlson, L. Carin, Preconditioned stochastic gradient langevin dynamics for deep neural networks, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[21] Y. W. Teh, A. H. Thiery, S. J. Vollmer, Consistency and fluctuations for stochastic gradient langevin dynamics, The Journal of Machine Learning Research 17 (2016) 193–225.

[22] A. G. Wilson, P. Izmailov, Bayesian deep learning and a probabilistic perspective of generalization, Advances in neural information processing systems 33 (2020) 4697–4708.

[23] A. Ashukha, A. Lyzhov, D. Molchanov, D. Vetrov, Pitfalls of in-domain uncertainty estimation and ensembling in deep learning, arXiv preprint arXiv:2002.06470 (2020).

[24] T. Duan, A. Anand, D. Y. Ding, K. K. Thai, S. Basu, A. Ng, A. Schuler, Ngboost: Natural gradient boosting for probabilistic prediction, in: International Conference on Machine Learning, PMLR, 2020, pp. 2690–2700.

[25] B. Settles, Active learning literature survey (2009).

[26] T. Tan, Z. Xiong, V. R. Dwaracherla, Parameterized indexed value function for efficient exploration in reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, 2020, pp. 5948–5955.

[27] S. Depeweg, J. M. Hernández-Lobato, F. Doshi-Velez, S. Udluft, Decomposition of uncertainty for active learning and reliable reinforcement learning in stochastic systems, stat 1050 (2017) 11.

[28] T. Garipov, P. Izmailov, D. Podoprikhin, D. P. Vetrov, A. G. Wilson, Loss surfaces, mode connectivity, and fast ensembling of dnns, Advances in neural information processing systems 31 (2018).

[29] F. Draxler, K. Veschgini, M. Salmhofer, F. Hamprecht, Essentially no barriers in neural network energy landscape, in: International conference on machine learning, PMLR, 2018, pp. 1309–1318.

[30] A. V. Dorogush, V. Ershov, A. Gulin, Catboost: gradient boosting with categorical features support, arXiv preprint arXiv:1810.11363 (2018).

[31] A. Malinin, N. Band, G. Chesnokov, Y. Gal, M. J. Gales, A. Noskov, A. Ploskonosov, L. Prokhorenkova, I. Provilkov, V. Raina, et al., Shifts: A dataset of real distributional shift across multiple large-scale tasks, arXiv preprint arXiv:2107.07455 (2021).

A. Comparison between cSGLB and cSG-MCMC

The proposed Cyclical SGLB algorithm combines SGLB with cSG-MCMC [13] to effectively explore different modes of a highly multimodal posterior distribution. In this section, we summarize some key differences between our design and the original cSG-MCMC algorithm.

(1) cSG-MCMC is a sampling algorithm designed for Bayesian NNs, while cSGLB is built for GBDT models. In deep learning, full-batch gradient descent is usually not feasible, and techniques have been developed to explicitly compensate mini-batch noises, such as preconditioning [14]. Some also suggested applying an additional correction step called Metropolis-Hastings [15]. Tree-based GB models can easily scale up to large industrial datasets and digest the full training set at each iteration. Therefore, our cSGLB uses full-batch GB in the sampling stage of each cycle to ensure high-quality samples being generated. (2) cSGLB puts a cyclical schedule on gradient scale while cSG-MCMC puts a schedule on step size. In addition, the original cSG-MCMC *completely removed* the injected Gaussian noises in the exploration stage, and cSG-MCMC reduced to regular stochastic gradient descent (SGD) during the period of exploration. Although the authors claimed that this amounts to posterior temping which is commonly used in DL domain, the implementation of cSG-MCMC algorithm does not follow closely/strictly the dynamics of SGLD during the exploration stage. In contract, we keep the injected noise term unchanged during the course of learning. Our design achieved similar effects compared with the step-size scheduling on a synthetic experiment, and we also ensure that cSGLB follows the dynamics of SGLD (more or less) at every iteration step. (3) Lastly, the gradient scaling (instead of step-size scheduling) has implementation benefits. The SGLB algorithm is made available in the CatBoost library [30], which only supports a constant step size (or learning rate). Our proposed cyclical gradient scaling (and data bootstrapping) can be implemented easily with the "user-defined loss function" available in the CatBoost package, without modifying a single line of the source code.

Data	# of samples						% of class		
	Total	Tropical	Dry	Mild	Snow	Polar	Class 0	Class 10	Class 20
train-ID	*200,000*	26,786	45,357	127,857	0	0	40.92	33.71	25.37
dev-ID	*46,279*	6,196	10,505	29,578	0	0	40.90	33.97	25.13
dev-OOD	*46,555*	0	0	0	46,555	0	38.35	35.07	26.58
eval-ID	*518,587*	69,245	117,981	331,361	0	0	40.98	33.71	25.30
eval-OOD	*524,048*	0	0	0	479,952	44,096	33.05	35.70	31.24

Table 2
Data summary of our partitioning of the Weather Prediction dataset.

B. Synthetic Data

B.1. Synthetic Multimodal Distribution

The ground truth density of the distribution is

$$F(x) = \sum_{i=1}^{25} \frac{1}{25} \mathcal{N}(x|\mu_i, \Sigma),$$

where $\mu = \{-4, -2, 0, 2, 4\}^T \times \{-4, -2, 0, 2, 4\}$ and $\Sigma = \begin{bmatrix} 0.03 & 0 \\ 0 & 0.03 \end{bmatrix}$. We used the code released by the authors [13] to generate our results. Specifically, the SGLD was trained with a decaying lr $\epsilon_\tau = 0.05(\tau + 1)^{-0.55}$, and clr-SGLD was learned with a cyclical lr schedule with initial value $\epsilon_0 = 0.09$ and exploration proportion $\eta = 0.25$. For our cg-SGLD, we fixed lr $\epsilon = 0.01$ and Gaussian noise scale as 0.4, and set $\alpha_{\max} = 10, \alpha_{\min} = 1$. The "noisy" version of SGLD (or NoisySGLD/N-SGLD) was trained with a fixed lr $\epsilon = 0.02$ and noise scale 5.0 (roughly $10\times$ larger than the noise scale used in the other methods). Each chain was trained for $50k$ iterations and both clr-SGLD and cg-SGLD had 30 cycles. The results and findings are robust to random seeds, and similar results were observed with different seeds. We refer the interested readers to the original paper [13] for results of SGLD and clr-SGLD in parallel (or multi-chain) settings.

B.2. Synthetic Spiral Dataset

All experiments were conducted using CatBoost [30], one of the-state-of-the-art libraries for GBDTs. The ensemble of SGLB (*ens*$_{20}$) contains 20 independent (with different random seeds) models with 1K trees each. The learning rate is $\epsilon = 0.1$, tree depth is 6, and *random_strength* $= 100$ and *border_count* $= 128$. The SGLB virtual ensemble and cSGLB virtual ensemble are trained with the same parameters except that we increase the number of trees to 2K and lower the lr for cSGLB to $\epsilon = 0.05$. Thus, the virtual ensemble is $10\times$ more efficient in computation and memory than the actual SGLB ensemble. For SGLB virtual ensemble, each 50th model from interval $[1000, 2000]$ is added to the ensemble, making it a virtual ensemble of 20 members. For cSGLB virtual ensemble, we set $\epsilon = 0.05$, cycle length $C = 200$, $\alpha_{\max} = 10, \alpha_{\min} = 1$, making it a virtual ensemble of $2000/200 = 10$ members. For cbSGLB with bootstrapping, we additionally set exploration proportion $\eta = 0.9$ and mask probability $p_{bm} = 0.66$.

C. Real-World Shifts Data

Data Summary A detailed summary of our final partitioning of the Weather Prediction dataset is included in Table 2.

Experimental Details We used default parameter settings for SGLB models as suggested in the original paper [12] for uncertainty quantification except that we set the subsample rate to 0.8 for stochastic gradient boosting. The real SGLB ensemble consists of (up to) 30 SGLB models trained with different seeds, each of 1K trees. In order to get more samples from a single chain, the virtual ensembles of SGLB and our cSGLB/cbSGLB were learned with a single model of 2K trees. We set learning rate for all models to $\epsilon = 0.05$, and tree depth to 6. For SGLB virtual ensemble, each 100th model from interval $[1000, 2000]$ was added to the ensemble, making it a virtual ensemble of 10 members. cSGLB and cbSGLB shared the same parameters with the SGLB counterpart. In addition, for cSGLB/cbSGLB virtual ensemble, we set cycle length $C = 200$, $\alpha_{\max} = 10.0, \alpha_{\min} = 1.0$, making it a virtual ensemble of $2000/200 = 10$ members. For simplicity, cSGLB used full-batch gradient boosting at each iteration step. In contrast, for cbSGLB with bootstrapping, we set exploration proportion $\eta = 0.8$, i.e., 80% of a cycle was treated as exploration, and set mask probability $p_{bm} = 0.6$ in the exploration stage. For model and parameter selection, we only used the in-domain (ID) development set and did not use the out-of-domain (OOD) development set. Although this may potentially lower our reported performance on the OOD evaluation set, we believe that it reflects better a real-world learning scenario where the shifted data is often unobserved and unavailable at training time.