Correcting optimiser Selection Bias via Large Deviation Hazards

Andrea Zerio [1 2 3] **Andres R. Masegosa** [3]

Abstract

Empirical risk minimisation systematically exploits finite-sample fluctuations of the training loss, producing the optimiser selection bias, responsible for miscalibration and generalisation failure in the interpolation regime. We introduce Stochastic Gradient Descent with Rejection sampling (SGDR), a drop-in modification to Stochastic Gradient Descent (SGD) that corrects this bias in loss space, with no change to the optimiser, architecture, or training objective. Each mini-batch is gated by how atypical its mean loss is under the population distribution, through a two-sided rejection rule derived from the hazard transform, with the population hazards estimated from held-out validation losses via rate functions from large deviation theory. We instantiate the hazards two ways: a scalar tail correction and a Bahadur-Rao (BR) saddlepoint correction. Across nine architectures spanning image classification on CIFAR-10/100 and graph classification on two molecular benchmarks, SGDR matches or improves on baseline performance while sharply reducing expected calibration error and overfitting, at a fraction of the training time and gradient updates required by standard SGD. The BR variant trades a small amount of accuracy for further calibration gains.

1. Introduction

Empirical risk minimisation selects parameters $\theta^\star$ where the training-loss distribution is artificially low relative to the population loss. In the overparameterised interpolation regime that now dominates deep learning (Zhang et al., 2021; Belkin et al., 2019), this *optimiser selection bias* is severe: training loss collapses toward zero while population loss retains irreducible spread, producing models that are systematically miscalibrated and overconfident (Guo et al., 2017).

We propose Stochastic Gradient Descent with Rejection sampling (SGDR), which corrects this bias at the batch level. Each mini-batch is scored by a two-sided penalty $\beta(B)$ measuring how atypical its mean loss is under the population distribution in either tail, and accepted with probability $e^{-\beta(B)}$ whilst rejected batches incur a forward pass but no gradient update. The penalty is constructed from batch-level deviation variables via the hazard transform and estimated in practice from validation losses using rate functions from large deviation theory (Dembo & Zeitouni, 2009).

In essence, SGDR is a drop-in modification to the SGD loop with a single correction-factor hyperparameter, and yields computational savings that scale with the severity of the mismatch it corrects. Despite still being early work, we make three contributions:

1. We formalise the optimiser selection bias in loss space and derive a two-sided rejection penalty that targets it via a distribution-free exponential (3);

2. We instantiate the penalty with a batch-average rate-function estimator and compare two variants differing in their treatment of the residual Bahadur-Rao gap (3.2);

3. We validate the method on synthetic classification, image classification on CIFAR-10/100, and graph classification on two molecular benchmarks (4). We show that SGDR consistently matches or improves on baseline predictive performance while sharply reducing expected calibration error and the generalisation gap, at a fraction of the wall-clock time and gradient updates required by standard SGD.

We present these as *early results*; a fuller evaluation with additional efficient-training baselines and gradient-level corrections is in preparation.

[1]Centre for Frontier AI Research, Agency for Science, Technology and Research, Singapore [2]Institute of High Performance Computing, Agency for Science, Technology and Research, Singapore [3]Department of Computer Science, Aalborg University, Copenhagen, Denmark. Correspondence to: Andres R. Masegosa <arma@cs.aau.dk>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

2. Background

Our method rests on two classical constructions that we recall here to fix notation and to motivate the mechanism developed in Section 3.

Empirical risk, population risk, and the interpolation regime. Let ν denote an unknown population distribution over pairs (Y, X) and $\mathcal{D} = \{(y_i, x_i)\}_{i=1}^n \sim \nu^n$ a training set. While we aim to minimise the *population risk* $\mathcal{R}(\theta) := \mathbb{E}_\nu[\ell(Y, X, \theta)]$, the standard paradigm is ERM: $\min_\theta \widehat{\mathcal{R}}_n(\theta) := \frac{1}{n}\sum_{i=1}^n \ell(y_i, x_i, \theta)$ (Vapnik, 1999). Let μ and $\widehat{\mu}$ denote the population and empirical laws of the random variable $Z := \ell(Y, X, \theta)$, respectively. In the overparameterised regime (Zhang et al., 2021; Belkin et al., 2019), the optimiser drives $\widehat{\mathcal{R}}_n \to 0$, causing $\widehat{\mu}_{\theta^\star}$ to concentrate at zero while $\mu_{\theta^\star}$ retains irreducible aleatoric spread. This discrepancy, which we term *optimiser selection bias*, is the statistical signature of interpolation rather than a failure mode. However, it leads to documented overconfidence and poor calibration (Guo et al., 2017). Correcting this bias without access to ν is our central objective.

Hazard Transform. For $Z \sim \mu$, let $S(a) := \mathbb{P}(Z \geq a)$ be the *survival function* and $G(a) := -\log S(a)$ the *cumulative hazard*. A specific case of the probability integral transform (Klein & Moeschberger, 2003) establishes that:

$$\alpha := G(Z) \sim \mathrm{Exp}(1), \tag{1}$$

independent of the functional form of μ. This identity provides a distribution-free target for assessing whether empirical samples follow the population law. Since $\widehat{\mu}$ is a biased proxy for μ under interpolation, Section 3 employs estimators of G derived from large deviation theory to construct a rejection sampler that thins training batches toward this unit exponential target.

3. Method

Correct sampling from the population loss law μ is equivalent, via (1), to producing loss values whose hazard-transformed counterparts follow $\mathrm{Exp}(1)$. We turn this equivalence into a training procedure in three steps: a two-sided penalty on batch-level deviations that quantifies how atypical a batch is relative to the population (3.1); a rate-function estimator for the population hazards, supplied by large deviation theory (3.2); and a rejection-sampling algorithm that gates the SGD backward pass on batch acceptance (3.3).

3.1. A two-sided penalty on batch deviations

Rejection sampling generates samples from a target distribution given only a proposal distribution (Robert et al., 2004): one draws a proposal, accepts it with a probability that corrects for the proposal-target mismatch, and retains the accepted draws. We apply this template in loss space, with the proposal given by standard mini-batch sampling and the target supplied by (1).

Fix a batch $\mathcal{B}$ of size B with mean loss

$$\bar{z}_\mathcal{B} := \frac{1}{B}\sum_{i\in\mathcal{B}} \ell(y_i, x_i, \theta), \qquad \bar{z} := \mathbb{E}_\nu[Z],$$

where $\bar{z}$ is the population mean at the current θ.

We define two non-negative batch-level deviation variables,

$$D_\mathcal{B}^- := \max(\bar{z} - \bar{z}_\mathcal{B},\, 0), \qquad D_\mathcal{B}^+ := \max(\bar{z}_\mathcal{B} - \bar{z},\, 0),$$

which measure the extent to which the batch mean falls below or above the population mean, respectively. At most one is non-zero.

A large D_B^- is the statistical signature of the optimiser selection bias: the model finds the batch unusually easy relative to the population. A large D_B^+ indicates the complementary pathology of outliers or label corruption. Applying the hazard construction of Section 2 separately to each deviation variable gives *deviation hazards*

$$G^-(d) := -\log \mathbb{P}(D^- \geq d),$$
$$G^+(\tilde{z}) := -\log \mathbb{P}(D^+ \geq \tilde{z}).$$

each of which inherits the property that under the population, $G^\pm(D^\pm) \sim \mathrm{Exp}(1)$. We combine them into a single non-negative penalty,

$$\beta(B) \;:=\; G^-(D_B^-) + G^+(D_B^+), \tag{2}$$

and accept batch B with probability $e^{-\beta(B)}$. The penalty vanishes on typical batches and grows with the log-probability that the population would produce a deviation at least as extreme as the observed one, in either tail. The acceptance rule thereby gates training on agreement between batch and population, treating left- and right-tail atypicality symmetrically at the level of β while leaving the asymmetry between pathologies to the shapes of G^- and G^+.

For a generic data statistic $g(Y, X)$, including the training gradient $g = \nabla_\theta \ell$, define the empirical and population conditional means

$$m(z) := \mathbb{E}_{\widehat{\nu}}[\, g \mid Z = z\,], \qquad m_\nu(z) := \mathbb{E}_\nu[\, g \mid Z = z\,].$$

The bias induced by replacing ν with $\widehat{\nu}$ decomposes into a loss-marginal mismatch and a conditional mismatch:

$$\mathbb{E}_{\widehat{\nu}}[g] - \mathbb{E}_\nu[g] = \underbrace{\int m(z)\,(\widehat{\mu} - \mu)(\mathrm{d}z)}_{\text{Term A: loss-marginal mismatch}} + \underbrace{\int \big(m(z) - m_\nu(z)\big)\,\mu(\mathrm{d}z)}_{\text{Term B: conditional mismatch}}. \tag{3}$$

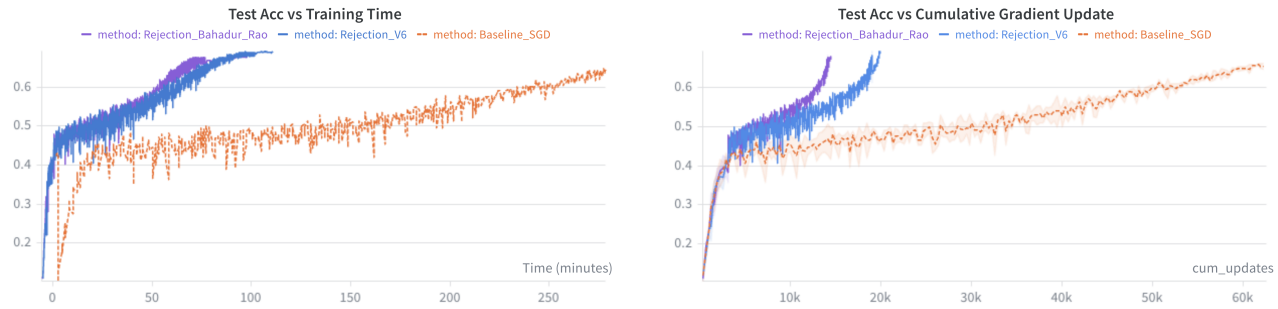

Figure 1. Efficiency gains demonstrated for a MobileNetV2 on CIFAR100. Rejection V6 refers to the standard configuration used throughout the paper. Full details available in Section C **Left:** test accuracy against training time. **Right:** test accuracy against cumulative gradient updates. Mean over three seeds; shaded region is ±one standard deviation.

When accepted batches have loss marginal μ, Term A vanishes exactly. Term A is the dominant contribution in the interpolation regime, driven by the collapse of $\widehat{\mu}_{\theta^\star}$ onto zero (Casado et al., 2024), whilst Term B persists. More details are provided in Section A.3.

3.2. Estimating the deviation hazards via rate functions

The population hazards $G^{\pm}$ are unavailable in practice. We estimate them using rate functions from large deviation theory (Dembo & Zeitouni, 2009), which give a smooth estimator that extrapolates reliably into the tails, where the penalty is informative and where empirical survival estimators are unstable. Let

$$\Lambda(\lambda) := \log \mathbb{E}_\nu\left[e^{\lambda(\bar{z}-Z)}\right], \qquad \Lambda(0) = \Lambda'(0) = 0,$$

denote the centred cumulant generating function (CGF) of the loss. The *rate functions* of the two deviations are its Legendre–Fenchel conjugates restricted to each tail,

$$I^-(d) := \sup_{\lambda\geq 0}\{\lambda d - \Lambda(\lambda)\}, \qquad I^+(\tilde{z}) := \sup_{\lambda\geq 0}\{\lambda \tilde{z} - \Lambda(-\lambda)\}.$$

Cramér's theorem identifies $I^{\pm}$ as the asymptotic exponential rates of the deviation probabilities and yields $I^- \leq G^-$ and $I^+ \leq G^+$ pointwise (Dembo & Zeitouni, 2009, Ch. 2).

The natural estimator of Λ from M individual validation losses fails in the tails: a single extreme loss dominates the exponential sum at large $|\lambda|$, collapsing the effective sample size where Λ is needed most. We circumvent this by estimating Λ from *batch averages*. From M held-out validation losses we form R batches of size B equal to the SGD batch size, compute their means $\bar{Z}_r$, and take

$$\widehat{\Lambda}_B(\lambda) := \log\left(\tfrac{1}{R}\sum_{r=1}^{R} e^{\lambda(\bar{z}_{\mathrm{ref}}-\bar{Z}_r)}\right), \tag{4}$$

with $\bar{z}_{\mathrm{ref}} := \frac{1}{M}\sum_m Z_m^{\mathrm{val}}$ as the estimator of $\bar{z}$. Three properties justify the choice: outlier losses are diluted by $1/B$ inside each batch mean; the CGF is estimated on the same scale at which the penalty is applied, requiring no rescaling; and batch averages have a smoother empirical distribution than individual losses, stabilising the Legendre-Fenchel transform numerically.

Two estimator variants. Cramér's theorem yields only the bound $I^{\pm} \leq G^{\pm}$ pointwise (Dembo & Zeitouni, 2009): substituting $\widehat{I}^{\pm}$ for the population hazards uniformly under-rejects atypical batches. We study two SGDR variants that handle the resulting gap differently.

The first, **SGDR**, sets $\widehat{G}_B^{\pm} = c^{\pm}\,\widehat{I}_B^{\pm}$ with scalar correction factors $c^{\pm} \geq 1$ applied per tail. The second, **SGDR (BR)**, closes the gap directly via the Bahadur–Rao saddlepoint correction (Bahadur & Rao, 1960),

$$\widehat{G}_B^{\mathrm{BR}}(d) := \widehat{I}_B(d) + \log\lambda^\star + \tfrac{1}{2}\log\Big(2\pi\,\widehat{\Lambda}_B''(\lambda^\star)\Big), \tag{5}$$

where $\lambda^\star = \arg\sup_{\lambda\geq 0}\{\lambda d - \widehat{\Lambda}_B(\lambda)\}$ is the saddlepoint attaining the supremum in I; the same correction applies to each tail. More details are provided in Section B.3. Comparing the two variants empirically tests whether closing the Cramér gap translates into measurable gains in generalisation and calibration; a full ablation of the rate-function estimator is provided in Appendix C.

3.3. Algorithm

The resulting training procedure intervenes minimally in the standard SGD loop. Each epoch begins with a single forward pass over the validation set (Phase 1), which produces an updated reference mean $\bar{z}_{\mathrm{ref}}^{(t)}$, the empirical batch-average CGF $\widehat{\Lambda}_{B,t}$, and its Legendre-Fenchel conjugates $\widehat{I}_{B,t}^{\pm}$ at the current model. Training then proceeds as standard SGD (Phase 2), except that each mini-batch is gated: after the forward pass we compute $\beta(B_t)$ via (2), draw $u \sim \mathrm{Unif}(0,1)$, and execute the backward pass only if $u \leq e^{-\beta(B_t)}$. Full pseudocode is given in Section B.1.

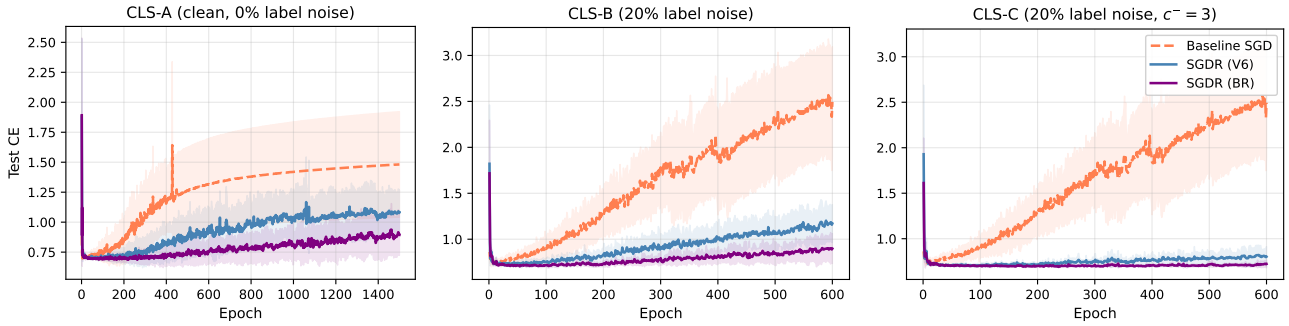

Figure 2. Test cross-entropy per epoch across the three synthetic configurations. SGDR V6 refers to the standard configuration used throughout the paper. More details in Section C. Mean over three seeds; shaded region is $\pm$ standard deviation.

Table 1. Image classification results on CIFAR-10 and CIFAR-100. Mean $\pm$ std over three seeds; **bold** marks the best per column within each architecture family.

		CIFAR-10			CIFAR-100		
Family	Method	Acc ↑	ECE ↓	Test CE ↓	Acc ↑	ECE ↓	Test CE ↓
MobileNetV2	Baseline SGD	88.74 ± 0.87	0.031 ± 0.002	0.346 ± 0.024	66.17 ± 1.24	0.074 ± 0.007	1.221 ± 0.035
	SGDR	90.33 ± 0.76	0.022 ± 0.001	0.294 ± 0.024	$\mathbf{67.41 \pm 1.08}$	0.033 ± 0.004	$\mathbf{1.125 \pm 0.046}$
	SGDR (BR)	$\mathbf{90.38 \pm 0.77}$	$\mathbf{0.016 \pm 0.001}$	$\mathbf{0.287 \pm 0.025}$	66.91 ± 1.56	$\mathbf{0.021 \pm 0.003}$	1.138 ± 0.058
ResNet	Baseline SGD	90.95 ± 0.59	0.047 ± 0.002	0.334 ± 0.007	65.14 ± 1.05	0.135 ± 0.024	1.448 ± 0.052
	SGDR	$\mathbf{91.26 \pm 0.64}$	0.025 ± 0.002	$\mathbf{0.267 \pm 0.017}$	$\mathbf{65.31 \pm 1.23}$	0.046 ± 0.003	$\mathbf{1.225 \pm 0.041}$
	SGDR (BR)	90.74 ± 0.46	$\mathbf{0.021 \pm 0.001}$	0.277 ± 0.014	64.16 ± 1.22	$\mathbf{0.039 \pm 0.003}$	1.255 ± 0.041
ShuffleNetV2	Baseline SGD	88.85 ± 1.53	0.042 ± 0.005	0.364 ± 0.030	$\mathbf{67.50 \pm 3.66}$	0.115 ± 0.015	1.290 ± 0.097
	SGDR	$\mathbf{89.80 \pm 1.89}$	0.026 ± 0.002	0.308 ± 0.051	66.19 ± 2.30	0.039 ± 0.002	$\mathbf{1.185 \pm 0.088}$
	SGDR (BR)	89.75 ± 1.72	$\mathbf{0.020 \pm 0.002}$	$\mathbf{0.307 \pm 0.050}$	65.13 ± 2.31	$\mathbf{0.030 \pm 0.001}$	1.218 ± 0.088
VGG	Baseline SGD	90.68 ± 0.81	0.059 ± 0.004	0.386 ± 0.029	$\mathbf{66.63 \pm 1.61}$	0.186 ± 0.010	1.707 ± 0.103
	SGDR	$\mathbf{91.13 \pm 0.86}$	0.032 ± 0.004	$\mathbf{0.285 \pm 0.027}$	65.69 ± 1.66	0.060 ± 0.005	$\mathbf{1.298 \pm 0.073}$
	SGDR (BR)	90.86 ± 0.99	$\mathbf{0.026 \pm 0.004}$	0.288 ± 0.028	64.30 ± 1.67	$\mathbf{0.048 \pm 0.004}$	1.329 ± 0.071

Computational cost. Let C_{fwd} be the cost of a forward pass and $C_{bwd} \approx 2C_{fwd}$ the cost of the corresponding backward pass, so that standard SGD costs $3C_{fwd}$ per batch. Under batch-level rejection with acceptance rate ρ, the per-batch cost is $(1 + 2\rho)C_{fwd}$, a fractional saving of $\frac{2}{3}(1 - \rho)$: 13% at $\rho = 0.8$, 33% at $\rho = 0.5$, 60% at $\rho = 0.1$.

Critically, the computational saving grows with the severity of the mismatch, so the method spends fewer gradient updates precisely on the batches whose gradients would bias the optimiser most.

4. Experiments

We evaluate SGDR in three stages: synthetic classification with controlled label noise (Section 4.1); image classification (Section 4.2); and graph classification (Section 4.3). Throughout, we compare Baseline SGD against SGDR and SGDR (BR), reporting mean and standard deviation over three seeds. We track the task-primary metric (accuracy or AUC), the expected calibration error (ECE), and test loss (CE or BCE). Full configurations and per-architecture tables are given in Section D.

4.1. Controlled synthetic validation

Three synthetic classification tasks isolate the tail-specific behaviour of the two-sided penalty. All share a three-class Gaussian mixture in $\mathbb{R}^{30}$ ($n = 150$) and a three-layer MLP, differing only in noise and correction factors: CLS-A is the standard classification set-up; CLS-B applies 20% label noise and uses symmetric scalar correction factors $c^- = c^+ = 1$; CLS-C keeps the noise but sets $c^- = 3$.

Figure 2 plots test cross-entropy per epoch and matches Section 3.1: on CLS-A the baseline overfits late while both SGDR variants hold the early minimum; on CLS-B SGDR matches or beats the baseline's best on the same corrupted data; on CLS-C asymmetric c^- compounds the protection. Both rejection variants are similar on CLS-A where the rate function is already tight; BR pulls ahead on CLS-B and CLS-C, where heavier tails widen the residual gap.

Table 2. Graph classification on two molecular benchmarks from the Open Graph Benchmark (Hu et al., 2020). Each row reports mean ± standard deviation across three seeds. Arrows indicate preferred direction; best method within each architecture is shown in bold.

		MolBACE			MolClinTox		
Family	Method	AUC ↑	ECE ↓	Test BCE ↓	AUC ↑	ECE ↓	Test BCE ↓
GCN	Baseline SGD	0.8091 ± 0.005	0.272 ± 0.007	1.109 ± 0.096	0.912 ± 0.020	0.125 ± 0.007	$\mathbf{0.522 \pm 0.085}$
	SGDR	0.826 ± 0.001	0.256 ± 0.039	$\mathbf{0.722 \pm 0.070}$	0.927 ± 0.009	0.074 ± 0.009	0.680 ± 0.098
	SGDR (BR)	$\mathbf{0.827 \pm 0.010}$	$\mathbf{0.251 \pm 0.042}$	0.743 ± 0.134	$\mathbf{0.934 \pm 0.011}$	$\mathbf{0.072 \pm 0.014}$	0.565 ± 0.053
GIN	Baseline SGD	0.790 ± 0.022	0.292 ± 0.003	1.200 ± 0.110	0.825 ± 0.011	0.169 ± 0.024	$\mathbf{0.728 \pm 0.027}$
	SGDR	0.797 ± 0.014	0.255 ± 0.007	$\mathbf{0.731 \pm 0.019}$	$\mathbf{0.8765 \pm 0.013}$	$\mathbf{0.120 \pm 0.015}$	0.7727 ± 0.060
	SGDR (BR)	$\mathbf{0.810 \pm 0.036}$	$\mathbf{0.251 \pm 0.048}$	0.773 ± 0.217	0.856 ± 0.014	0.129 ± 0.026	0.993 ± 0.069
SAGE	Baseline SGD	0.770 ± 0.008	0.283 ± 0.020	1.009 ± 0.072	0.896 ± 0.004	0.195 ± 0.049	0.536 ± 0.029
	SGDR	0.790 ± 0.011	$\mathbf{0.225 \pm 0.013}$	$\mathbf{0.692 \pm 0.042}$	$\mathbf{0.926 \pm 0.002}$	$\mathbf{0.106 \pm 0.018}$	0.573 ± 0.017
	SGDR (BR)	$\mathbf{0.804 \pm 0.009}$	0.250 ± 0.054	0.729 ± 0.136	0.921 ± 0.001	0.124 ± 0.017	$\mathbf{0.511 \pm 0.021}$
GT	Baseline SGD	0.787 ± 0.004	$\mathbf{0.257 \pm 0.015}$	0.836 ± 0.028	0.891 ± 0.001	0.274 ± 0.021	0.572 ± 0.062
	SGDR	0.786 ± 0.020	0.282 ± 0.046	$\mathbf{0.821 \pm 0.108}$	0.909 ± 0.006	$\mathbf{0.128 \pm 0.021}$	0.589 ± 0.026
	SGDR (BR)	$\mathbf{0.795 \pm 0.028}$	0.313 ± 0.031	0.897 ± 0.086	$\mathbf{0.921 \pm 0.014}$	0.188 ± 0.024	$\mathbf{0.451 \pm 0.050}$
GaTv2	Baseline SGD	$\mathbf{0.807 \pm 0.003}$	0.314 ± 0.011	0.951 ± 0.025	0.929 ± 0.002	0.181 ± 0.026	$\mathbf{0.419 \pm 0.063}$
	SGDR	0.800 ± 0.021	0.255 ± 0.036	0.741 ± 0.097	$\mathbf{0.946 \pm 0.008}$	$\mathbf{0.087 \pm 0.002}$	0.422 ± 0.084
	SGDR (BR)	0.795 ± 0.014	$\mathbf{0.236 \pm 0.024}$	$\mathbf{0.700 \pm 0.062}$	0.936 ± 0.019	0.098 ± 0.013	0.461 ± 0.171

4.2. Image classification

We train fifteen model variants spanning 4 architecture classes: ResNet (He et al., 2016); VGG (Simonyan & Zisserman, 2014); MobileNetV2 (Sandler et al., 2018) and ShuffleNetv2 (Ma et al., 2018); on CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009). Table 1 aggregates results to the family level; per-architecture breakdowns are provided in Section D.2.

Rejection sampling consistently yields models that are comparable to or exceed the baseline in accuracy while sharply improving calibration and test Cross Entropy (CE). ECE and Test CE fall across every family and dataset, often by a factor of two or more, validating the distributional claim of Section 3.1: correcting $\widehat{\mu}_{\theta^\star}$ toward $\mu_{\theta^\star}$ yields measurably better-calibrated models. The BR variant trades a small amount of task accuracy for additional calibration and generalisation gains, allowing users to control the trade-off between performance and calibration depending on the task.

4.3. Graph classification

We evaluate five GNN architectures: GCN (Kipf & Welling, 2016); GIN (Xu et al., 2018); GraphSAGE (Hamilton et al., 2017); GraphTransformer (Yun et al., 2019) and GATv2 (Brody et al., 2021); on two molecular binary classification benchmarks from the Open Graph Benchmark (Hu et al., 2020), MolBACE and MolClinTox. The complete experimental setup is provided in Section D.1.

The results in Table 2 show that the pattern from Section 4.2 transfers to graph data. On MolClinTox, SGDR roughly halves ECE on four of five architectures and improves AUC across all five. On MolBACE, Test BCE drops by roughly a quarter across the board and AUC improves on four of five architectures.

4.4. Efficiency

Figure 1 reports test accuracy of MobileNetV2 on CIFAR-100 against wall-clock training time (left) and cumulative gradient updates (right). Both SGDR variants reach any given performance level in a fraction of the time and updates required by the baseline, realising the per-batch cost saving described in Section 3.3.

5. Conclusion and Future Work

We introduced SGDR, a training procedure that mitigates the optimiser selection bias of ERM by thinning mini-batches with a two-sided penalty derived from the hazard transform and large deviation rate functions. The penalty targets the loss-marginal component of the bias (Term A), which dominates in the interpolation regime, with the population hazards estimated either by a scalar tail correction or by a Bahadur-Rao saddlepoint correction.

Across nine image and graph architectures, both variants reduce calibration error and generalisation gaps at lower computational cost than standard ERM, and together they trace a calibration-accuracy frontier, with the saddlepoint correction better calibrated and the scalar correction retaining a small task-performance edge. Future work will focus on (i) benchmarking against efficient-training baselines and (ii) developing gradient-level corrections for the residual conditional mismatch (Term B), which loss-marginal matching alone leaves untouched.

References

Bahadur, R. R. and Rao, R. R. On deviations of the sample mean. *The Annals of Mathematical Statistics*, 31(4):1015–1027, 1960.

Belkin, M., Hsu, D., Ma, S., and Mandal, S. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

Brody, S., Alon, U., and Yahav, E. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

Casado, I., Ortega, L. A., Pérez, A., and Masegosa, A. R. Pac-bayes-chernoff bounds for unbounded losses. *Advances in Neural Information Processing Systems*, 37: 24350–24374, 2024.

Dembo, A. and Zeitouni, O. *Large deviations techniques and applications*, volume 38. Springer Science & Business Media, 2009.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International conference on machine learning*, pp. 1321–1330. PMLR, 2017.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Klein, J. P. and Moeschberger, M. L. *Survival analysis: techniques for censored and truncated data*, volume 1230. Springer, 2003.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116–131, 2018.

Robert, C. P., Casella, G., and Casella, G. *Monte Carlo statistical methods*, volume 2. Springer, 2004.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

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

Vapnik, V. N. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018.

Yun, S., Jeong, M., Kim, R., Kang, J., and Kim, H. J. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

Impact Statement

This work advances the efficiency and reliability of deep learning models, contributing to the development of more accessible and robust machine learning systems across diverse applications.

A. Extended Background

A.1. PAC-Bayes-Chernoff Bounds and the Centred CGF $\Lambda_\theta(\lambda)$

The centred cumulant generating function (CGF)

$$\Lambda_\theta(\lambda) \;:=\; \log \mathbb{E}_\nu\Big[e^{\lambda(\bar{z}-Z)}\Big], \qquad \Lambda_\theta(0) = \Lambda'_\theta(0) = 0, \tag{6}$$

that defines our estimator in (4) is the same object that appears in the recent PAC-Bayes-Chernoff framework for unbounded losses of Casado et al. (2024). Their bound takes the form

$$\mathbb{E}_{\theta\sim\rho}[\mathcal{R}(\theta)] \;\le\; \mathbb{E}_{\theta\sim\rho}[\widehat{\mathcal{R}}_n(\theta)] \;+\; (I_\rho)^{-1}\left(\frac{\mathrm{KL}(\rho\,\|\,\pi) + \log(1/\delta)}{n}\right), \tag{7}$$

with probability at least $1-\delta$ over $\mathcal{D} \sim \nu^n$, where I_ρ^{-1} is the inverse of the Cramér rate function obtained from Λ_θ averaged over the posterior ρ. The inverse rate function plays the role of a complexity penalty whose magnitude scales with the heaviness of the loss tail.

Two implications for the present work. First, the same Λ_θ that controls generalisation in (7) is the object we estimate from validation losses to construct the deviation hazards $\widehat{G}^{\pm}$ and the rejection penalty β. Our method does not introduce new statistical machinery; it repurposes an established generalisation-theoretic object as a sampling correction. Second, (7) is a posterior-level statement in ρ, whereas our use of Λ_θ is parameter-pointwise, evaluated at the current iterate $\theta^{(t)}$ during training. The two uses are mathematically the same but operationally distinct: the bound certifies generalisation *after* training, whereas the rejection rule corrects the empirical loss law *during* training.

A.2. Hazard transform identity proof

We restate and prove the hazard identity (1). Let Z be a non-negative random variable with continuous law μ, survival function $S(a) = \mathbb{P}(Z \ge a)$, and cumulative hazard $G(a) = -\log S(a)$. Define $\alpha := G(Z)$.

Proposition A.1 (Hazard identity)**.** $\alpha \sim \mathrm{Exp}(1)$*, regardless of the form of* μ*.*

Proof. G is non-decreasing, so for any $t \ge 0$,

$$\mathbb{P}(\alpha \ge t) \;=\; \mathbb{P}(G(Z) \ge t) \;=\; \mathbb{P}\big(Z \ge G^{-1}(t)\big) \;=\; S\big(G^{-1}(t)\big) \;=\; e^{-G(G^{-1}(t))} \;=\; e^{-t},$$

which is the survival function of the unit exponential. The converse follows from the same calculation in reverse: if $\alpha \sim \mathrm{Exp}(1)$, then $Z = G^{-1}(\alpha) \sim \mu$. □

Two consequences are used in Section 3. First, the identity supplies a distribution-free target: a sample drawn from μ, passed through G, must look unit exponential. Second, applying Proposition A.1 separately to the deviation variables D^- and D^+ of Section 3.1 yields $G^{\pm}(D^{\pm}) \sim \mathrm{Exp}(1)$ under the population.

A.3. Bias decomposition

We derive the loss-marginal / conditional decomposition asserted in Section 3.1. Let $g(Y, X)$ be a generic data statistic; in our setting, $g = \nabla_\theta \ell(Y, X, \theta)$. Define the conditional expectations

$$m_\nu(z) \;:=\; \mathbb{E}_\nu[g(Y,X) \mid Z = z], \qquad m(z) \;:=\; \mathbb{E}_{\widehat{\nu}}[g(Y,X) \mid Z = z],$$

the population and empirical means of g given that the loss equals z. By the law of total expectation,

$$\mathbb{E}_\nu[g] \;=\; \int m_\nu(z)\,\mu(\mathrm{d}z), \qquad \mathbb{E}_{\widehat{\nu}}[g] \;=\; \int m(z)\,\widehat{\mu}(\mathrm{d}z).$$

Adding and subtracting $\int m(z)\,\mu(\mathrm{d}z)$ in the difference yields

$$\mathbb{E}_{\widehat{\nu}}[g] - \mathbb{E}_{\nu}[g] \;=\; \underbrace{\int m(z)\,(\widehat{\mu}-\mu)(\mathrm{d}z)}_{\text{Term A: loss-marginal mismatch}} \;+\; \underbrace{\int (m(z)-m_{\nu}(z))\,\mu(\mathrm{d}z)}_{\text{Term B: conditional mismatch}}. \tag{8}$$

Term A vanishes under exact loss-marginal matching. If accepted batches have loss marginal μ exactly, then $\widehat{\mu}=\mu$ and Term A is identically zero. In practice, the matching is approximate and the residual scales with the tightness of the rate-function estimator at the deviations the sampler encounters. Term B persists as a residual that depends on the conditional distribution of g given Z. In the interpolation regime, Term A dominates because $\widehat{\mu}_{\theta^\star}$ collapses onto zero while $\mu_{\theta^\star}$ retains spread (Casado et al., 2024). Term B captures a subtler effect: two different inputs can produce the same loss value with different gradients. Loss-marginal matching ensures $\widehat{\mu}=\mu$, but does not control which specific inputs land at each loss level, so the gradient directions averaged at fixed z may still differ between $\widehat{\nu}$ and ν. Term B is this residual, and addressing it requires gradient-level corrections (Section 5).

B. Algorithm Details

B.1. SGDR Pseudocode

Algorithm 1 SGDR: batch-level rejection SGD

Require: Training set $\mathcal{D}$, validation set of size M, batch size B, initial parameters θ_0, learning rate η, epochs T.

Initialisation. Run Phase 1 on the validation set to obtain $\bar{z}_{\mathrm{ref}}^{(0)}$, $\widehat{\Lambda}_{B,0}$, and the deviation-hazard estimators $\widehat{G}^{\pm}_{B,0}$.
for $t = 1,\dots,T$ **do**
  Phase 1: validation pass.
    Forward pass on the validation set; form R batches of size B; update $\widehat{\Lambda}_{B,t}$, $\widehat{G}^{\pm}_{B,t}$, $\bar{z}_{\mathrm{ref}}^{(t)}$.
  Phase 2: training with rejection.
  for each mini-batch B_t drawn from $\mathcal{D}$ **do**
    Forward pass; compute $\bar{z}_{B_t}$.
    $\beta(B_t) \leftarrow \widehat{G}^{-}_{B,t}\Big(\bar{z}_{\mathrm{ref}}^{(t)} - \bar{z}_{B_t}\Big) + \widehat{G}^{+}_{B,t}\Big(\bar{z}_{B_t} - \bar{z}_{\mathrm{ref}}^{(t)}\Big)$
    Draw $u \sim \mathrm{Unif}(0,1)$.
    if $u \le \exp(-\beta(B_t))$ **then**
      Backward pass; $\theta \leftarrow \theta - \eta\nabla_\theta \widehat{\mathcal{R}}_{B_t}(\theta)$.
    end if
  end for
end for

Variant instantiation. For SGDR: $\widehat{G}^{\pm}_{B,t}(\cdot) = c^{\pm}\,\widehat{I}^{\pm}_{B,t}(\cdot)$ with $c^{\pm}=1$ throughout (Section 3.2). For SGDR (BR): $\widehat{G}^{\pm}_{B,t}$ is the Bahadur-Rao corrected estimator of (5).

B.2. Batch-average CGF estimation and Legendre–Fenchel transform

The batch-average CGF estimator (4) is

$$\widehat{\Lambda}_B(\lambda) \;=\; \log\left(\frac{1}{R}\sum_{r=1}^{R} e^{\lambda(\bar{z}_{\mathrm{ref}}-\bar{Z}_r)}\right). \tag{9}$$

We discuss three implementation choices that the ablation in Section C varies: numerical stability, batch construction, and the Legendre–Fenchel solver.

Numerical stability. The exponential sum is computed via log-sum-exp:

$$\widehat{\Lambda}_B(\lambda) \;=\; \mathrm{LSE}_r\big\{\lambda(\bar{z}_{\mathrm{ref}}-\bar{Z}_r)\big\} - \log R, \qquad \mathrm{LSE}_r\{x_r\} := \max_r x_r + \log\sum_r e^{x_r - \max_r x_r},$$

which avoids overflow at large $|\lambda|$ where the dominant term in the sum is exponentially larger than the rest.

Batch construction. Given M validation losses, $\widehat{\Lambda}_B$ requires R batch averages of size B. We compare three strategies:

- **Non-overlapping.** Partition the M losses into $R = \lfloor M/B \rfloor$ disjoint groups of size B. Each $\bar{Z}_r$ is an i.i.d. sample from the law of $\bar{Z} = \frac{1}{B}\sum_{i=1}^{B} Z_i$, matching the assumptions of Cramér's theorem exactly. Sample size R is constrained by M.
- **Bootstrap.** Sample R batches of size B from the M validation losses with replacement. Sample size R is decoupled from M, but successive batch averages are correlated, biasing $\widehat{\Lambda}_B$ toward the empirical CGF of the validation set rather than the population CGF.
- **Overlap.** Use a sliding window of size B over the M losses with stride 1, giving $R = M - B + 1$ batches. Highly correlated successive averages, but maximises the number of distinct batch realisations.

Legendre–Fenchel transform. The rate function $\widehat{I}^{-}(d) = \sup_{\lambda \geq 0}\{\lambda d - \widehat{\Lambda}_B(\lambda)\}$ is computed by solving for the supremum at each query point d. The objective $\phi_d(\lambda) := \lambda d - \widehat{\Lambda}_B(\lambda)$ is concave in λ on the effective domain of $\widehat{\Lambda}_B$, since $\widehat{\Lambda}_B$ is convex by construction. We compare two solvers:

- **Grid search.** Evaluate ϕ_d at K uniformly spaced points $\lambda \in [0, \lambda_{\max}]$ and return the maximum. Cost is $O(KR)$ per evaluation; precision is $O(\lambda_{\max}/K)$.
- **Ternary search.** Bracket the maximum and bisect repeatedly using ternary subdivision, exploiting concavity. Cost is $O(R\log(1/\epsilon))$ for tolerance ϵ; for our default settings, ~ 30 iterations suffice.

The right-tail estimator $\widehat{I}^{+}(\tilde{z}) = \sup_{\lambda \geq 0}\{\lambda \tilde{z} - \widehat{\Lambda}_B(-\lambda)\}$ is computed analogously. Empirically (Section C), ternary search produces tighter estimates than grid search at comparable wall-clock cost, and is the default in all main-body experiments.

B.3. Bahadur–Rao saddlepoint correction derivation

Cramér's theorem provides the upper bound $\mathbb{P}(\bar{Z} - \bar{z} \geq d) \leq e^{-RI(d)}$ for the empirical mean of an i.i.d. sample of size R. The Bahadur–Rao theorem (Bahadur & Rao, 1960) sharpens this to an asymptotic equality with a multiplicative correction:

$$\mathbb{P}(\bar{Z} - \bar{z} \geq d) \;=\; \frac{1}{\lambda^\star \sqrt{2\pi R\, \Lambda''(\lambda^\star)}}\, e^{-RI(d)} \left(1 + O(R^{-1})\right), \tag{10}$$

where $\lambda^\star = \arg\sup_{\lambda \geq 0}\{\lambda d - \Lambda(\lambda)\}$ is the saddlepoint and $\Lambda''(\lambda^\star)$ is the second derivative of the CGF at the saddlepoint. Taking the negative logarithm and dividing by R,

$$-\frac{1}{R}\log \mathbb{P}(\bar{Z} - \bar{z} \geq d) \;=\; I(d) + \frac{1}{R}\left[\log \lambda^\star + \tfrac{1}{2}\log(2\pi R\, \Lambda''(\lambda^\star))\right] + O(R^{-2}).$$

The left-hand side is the cumulative hazard G of the batch-mean deviation, divided by R. Evaluating at the per-batch scale ($R = 1$) and absorbing the $\log R$ term into the additive constant, we obtain

$$\widehat{G}^{\mathrm{BR}}(d) \;=\; \widehat{I}(d) + \log \lambda^\star + \tfrac{1}{2}\log\Big(2\pi\, \widehat{\Lambda}''_B(\lambda^\star)\Big), \tag{11}$$

which is (5). The correction is asymptotically exact to order $O(R^{-1})$ (Bahadur & Rao, 1960).

Computational cost. The saddlepoint $\lambda^\star$ is already computed during the Legendre–Fenchel transform; the second derivative $\widehat{\Lambda}''_B(\lambda^\star)$ is obtained by direct analytic differentiation of (9) (a weighted second moment of the batch-mean deviations under the tilted measure) at the cost of one additional pass over the R batch means. The total overhead of BR over the standard SGDR is therefore a small constant per query.

B.4. Effective Sample Size $R_{\mathrm{eff}}(\lambda)$ and λ-Grid Truncation

The estimator (9) is a self-normalised importance-sampling sum with weights $w_r(\lambda) = e^{\lambda(\bar{z}_{\mathrm{ref}} - \bar{Z}_r)}$. Its effective sample size,

$$R_{\mathrm{eff}}(\lambda) \;:=\; \frac{\left(\sum_r w_r(\lambda)\right)^2}{\sum_r w_r(\lambda)^2}, \tag{12}$$

satisfies $R_{\mathrm{eff}}(0) = R$ (uniform weights) and $R_{\mathrm{eff}}(\lambda) \to 1$ as $|\lambda|$ grows, when one weight dominates the sum.

Stability region and truncation. We define the *stable region* as $\Lambda_{\text{stab}} := \{\lambda : R_{\text{eff}}(\lambda) \geq \tau R\}$ for a threshold $\tau \in (0, 1]$ (default $\tau = 0.1$). The Legendre–Fenchel supremum is restricted to Λ_{stab}:

$$\widehat{I}^{\pm}_{\text{trunc}}(d) \;=\; \sup_{\lambda \in \Lambda_{\text{stab}},\, \lambda \geq 0} \big\{\lambda d - \widehat{\Lambda}_B(\pm\lambda)\big\}.$$

This has two practical consequences. First, very large deviations d, which would require a saddlepoint $\lambda^\star$ outside the stable region, are clipped to the largest reliably-estimated deviation $d^\star$; for $d > d^\star$ the estimator is held constant at the boundary value, a conservative extrapolation. Second, the saddlepoint used in the BR correction (11) is constrained to Λ_{stab}. For deviations within the support of the validation distribution, this constraint is rarely active.

Effect of λ range. The choice of nominal λ range interacts with R_{eff}. A narrow range (e.g. $\lambda \in [0, 10]$) effectively imposes a fixed truncation regardless of R_{eff} and tends to underestimate the rate function in the tails. A wide range (e.g. $\lambda \in [0, 100]$ or larger) combined with adaptive R_{eff} truncation produces tighter estimates without numerical instability. Section C reports both regimes empirically.

C. Rate-Function Estimator Ablation

The penalty in (2) requires reliable estimates of the deviation hazards $G^{\pm}$. Three implementation choices identified in Section B.2 interact non-trivially with rate-function quality and downstream task performance: the Legendre–Fenchel solver (grid search vs. ternary), the batch-construction strategy (non-overlapping, bootstrap, overlap windows), and the λ range. We test nine variants V0–V8 spanning these axes (Table 3), and additionally compare the default V6 against the Bahadur-Rao corrected estimator of (5) and an oracle Monte Carlo estimator.

All experiments share the synthetic setting of Section 4.1 (three-class Gaussian mixture in $\mathbb{R}^{30}$, $n = 150$, three-layer MLP with hidden width 256). Ablation experiments are trained for 600 epochs with batch size 8 and learning rate 0.1 over 3 seeds. We evaluate two configurations: *CLS-A* (no additional noise, $c^- = 1$) and *CLS-C* (20% label noise, $c^- = 3$). Variants are estimated from $M = 300$ validation losses; oracle ground-truth estimates use $M = 50{,}000$ Monte Carlo samples. Figures 4–5 compare both Rejection V6 and BR variants estimators against the oracle Monte Carlo estimate at θ_{mid}. They additionally include a CLT Gaussian reference, which approximates the deviation log-survival as $-\log \Pr(D \geq d) \approx d^2/(2\sigma^2/B)$ using the validation-set variance σ^2 and batch size B; this is the second-order Taylor expansion of Λ at zero, and serves as a baseline for what a Gaussian-tail assumption would predict. Rate-function quality is assessed at the 150-epoch snapshot θ_{mid}. Figures 6–7 report the corresponding downstream task performance.

Table 3. Nine variants of the rate-function estimator. V0–V2 sweep grid resolution and λ range under non-overlapping batches; V3–V4 sweep λ range under ternary search; V5–V8 cross batch construction (bootstrap, overlap) with solver (grid, ternary). The default used throughout the main body is V6.

ID	Configuration
V0	Grid search, $K = 300$, $\lambda \in [0, 10]$ (non-overlapping batches)
V1	Grid search, $K = 300$, $\lambda \in [0, 100]$ (non-overlapping batches)
V2	Grid search, $K = 3000$, $\lambda \in [0, 100]$ (non-overlapping batches)
V3	Ternary search, $\lambda \in [0, 100]$ (non-overlapping batches)
V4	Ternary search, $\lambda \in [0, 10^6]$ (non-overlapping batches)
V5	Bootstrap batches, $R = 500$, grid search
V6	**Bootstrap batches,** $R = 500$**, ternary search**
V7	Overlap windows, grid search
V8	Overlap windows, ternary search

Why V6. The variants cluster tightly on the downstream metric (Figure 3, right), but V6 attains the largest improvement over vanilla within the same wall-clock budget. Three properties motivate the V6 default. First, ternary search produces tighter rate-function estimates than grid search at comparable cost, exploiting the concavity of the Legendre-Fenchel objective rather than relying on grid resolution. Second, among ternary variants (V3, V4, V6, V8), bootstrap batches most effectively reduce variance in the CGF estimate $\widehat{\Lambda}_B$: resampling with replacement decouples the effective sample size from the validation set size M, producing smoother CGF curves and more stable rate-function estimates. Third, the wide

$\lambda \in [0, 10^6]$ range combined with adaptive R_{eff} truncation (Section B.4) is necessary to resolve the sharp tail behaviour the rejection penalty depends on; narrow-range variants (V0, $\lambda_{\max} = 10$) systematically underestimate the rate function in the tails.

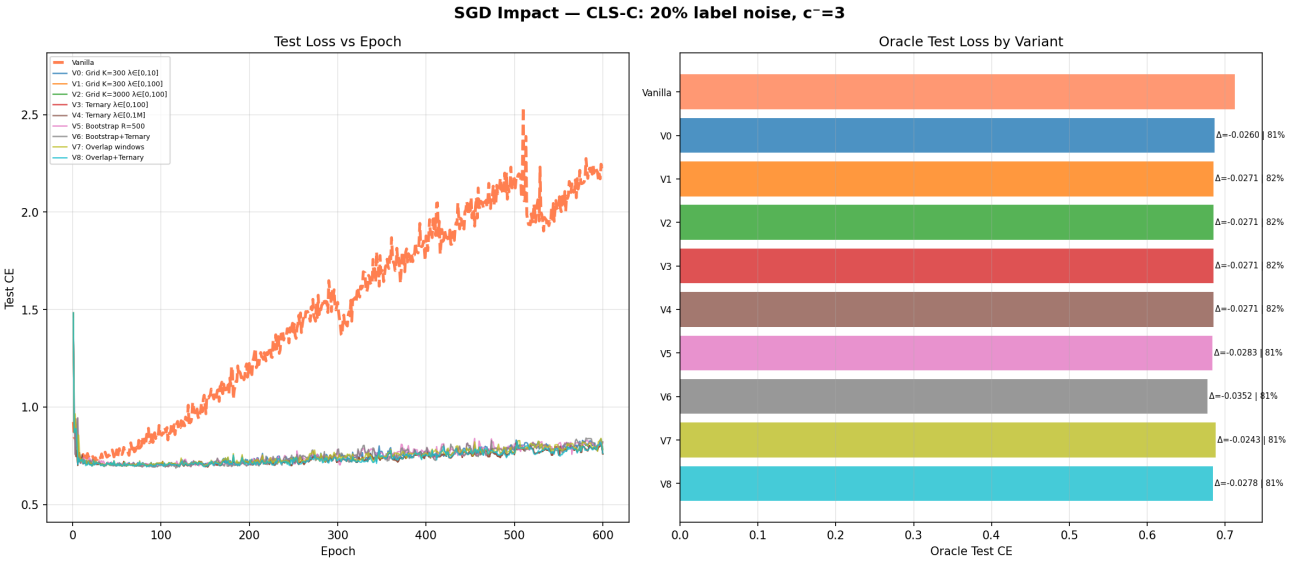

Figure 3. Downstream impact of the nine variants on CLS-C (20% label noise, $c^- = 3$). **Left:** test cross-entropy per epoch for vanilla SGD and the nine SGDR variants. Vanilla overfits the corrupted labels monotonically; all SGDR variants hold the early minimum throughout training. **Right:** oracle test CE at the best epoch per run, with absolute change Δ relative to vanilla and the per-batch wall-clock saving. All variants improve over vanilla by $\Delta \in [-0.0352, -0.0243]$ at 81–82% wall-clock saving, and V6 (gray) attains the largest Δ.

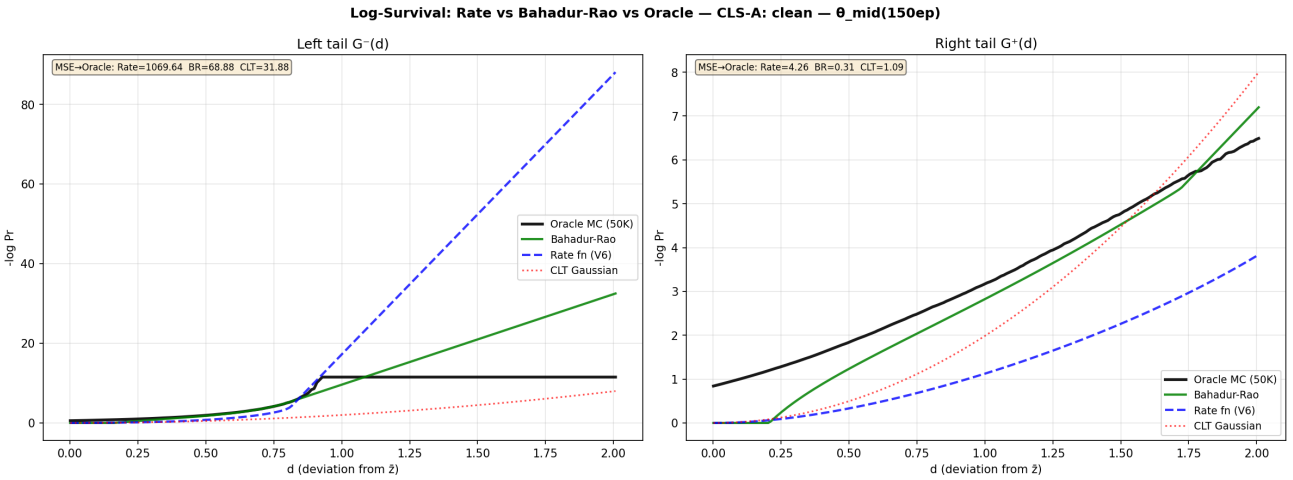

Figure 4. Log-survival functions on CLS-A (clean) at θ_{mid} (150 epochs). **Left:** left tail $G^-(d)$. **Right:** right tail $G^+(d)$. The bare rate function (V6, blue dashed) systematically overestimates the left-tail log-survival and underestimates the right tail; the Bahadur-Rao correction (green) reduces both gaps substantially. The CLT Gaussian approximation (red dotted) is included as a reference. MSE relative to oracle (legend boxes): on the left tail, Rate $= 1069.64$, BR $= 68.88$, CLT $= 31.88$; on the right tail, Rate $= 4.26$, BR $= 0.31$, CLT $= 1.09$.

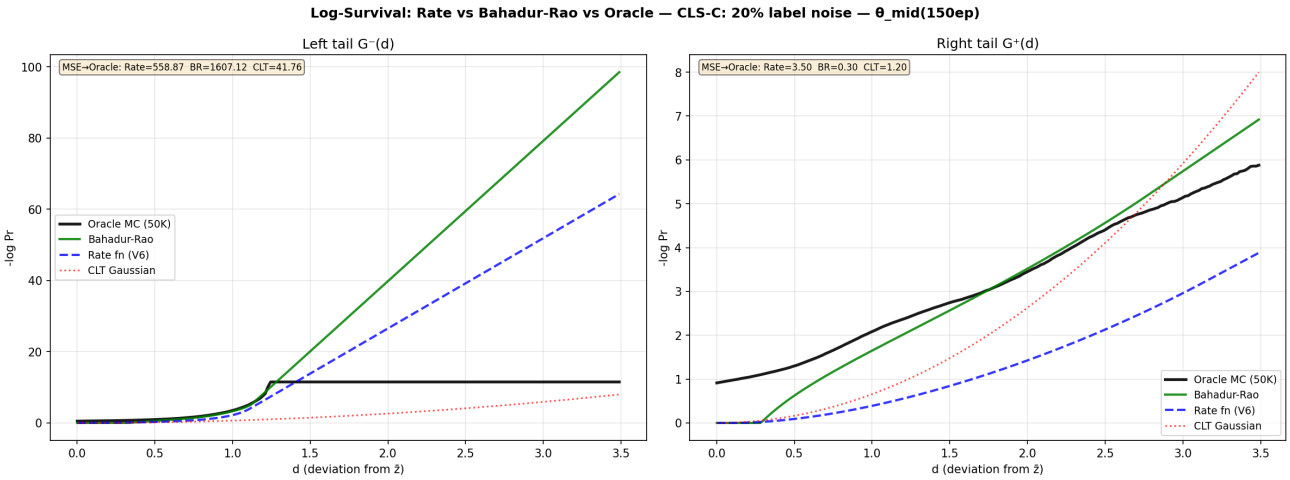

Figure 5. Log-survival functions on CLS-C (20% label noise) at θ_{mid} (150 epochs). On the right tail, where label noise concentrates outlier mass, BR closes most of the V6-to-oracle gap (Rate $= 3.50$, BR $= 0.30$, CLT $= 1.20$). On the left tail, BR grows faster than V6 beyond $d \approx 1.25$, where the oracle Monte Carlo estimate has plateaued at $-\log \Pr \approx 12$ owing to the resolution limit of $M = 50{,}000$ samples; this drives BR's left-tail MSE above V6's at θ_{mid} (Rate $= 558.87$, BR $= 1607.12$). This upward push that BR's saddlepoint correction applies in the deep tails translates into more aggressive rejection of atypically easy batches under noise. This is the mechanism behind the V6/BR trade-off observed in Table 1 and Table 2, where BR is consistently better-calibrated while V6 retains a small accuracy edge.

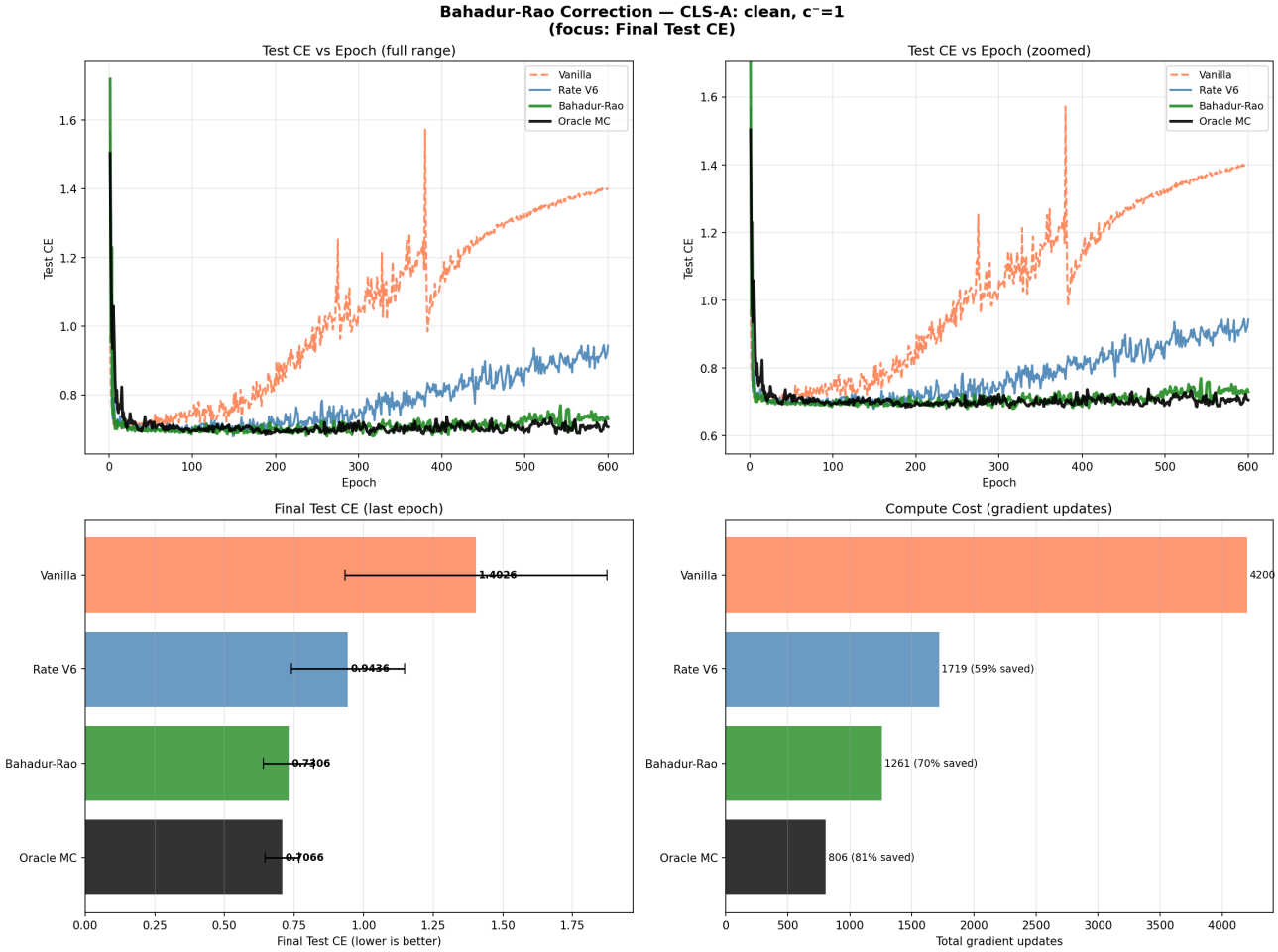

Figure 6. Saddlepoint correction on CLS-A. Vanilla SGD overfits monotonically (final test CE $= 1.4026$). The bare rate function (V6) plateaus near 0.94, the Bahadur-Rao correction reaches 0.73, and the oracle Monte Carlo estimator reaches 0.71. BR closes most of the gap between V6 and the oracle while saving 70% of gradient updates; V6 saves 59%, oracle saves 81%.

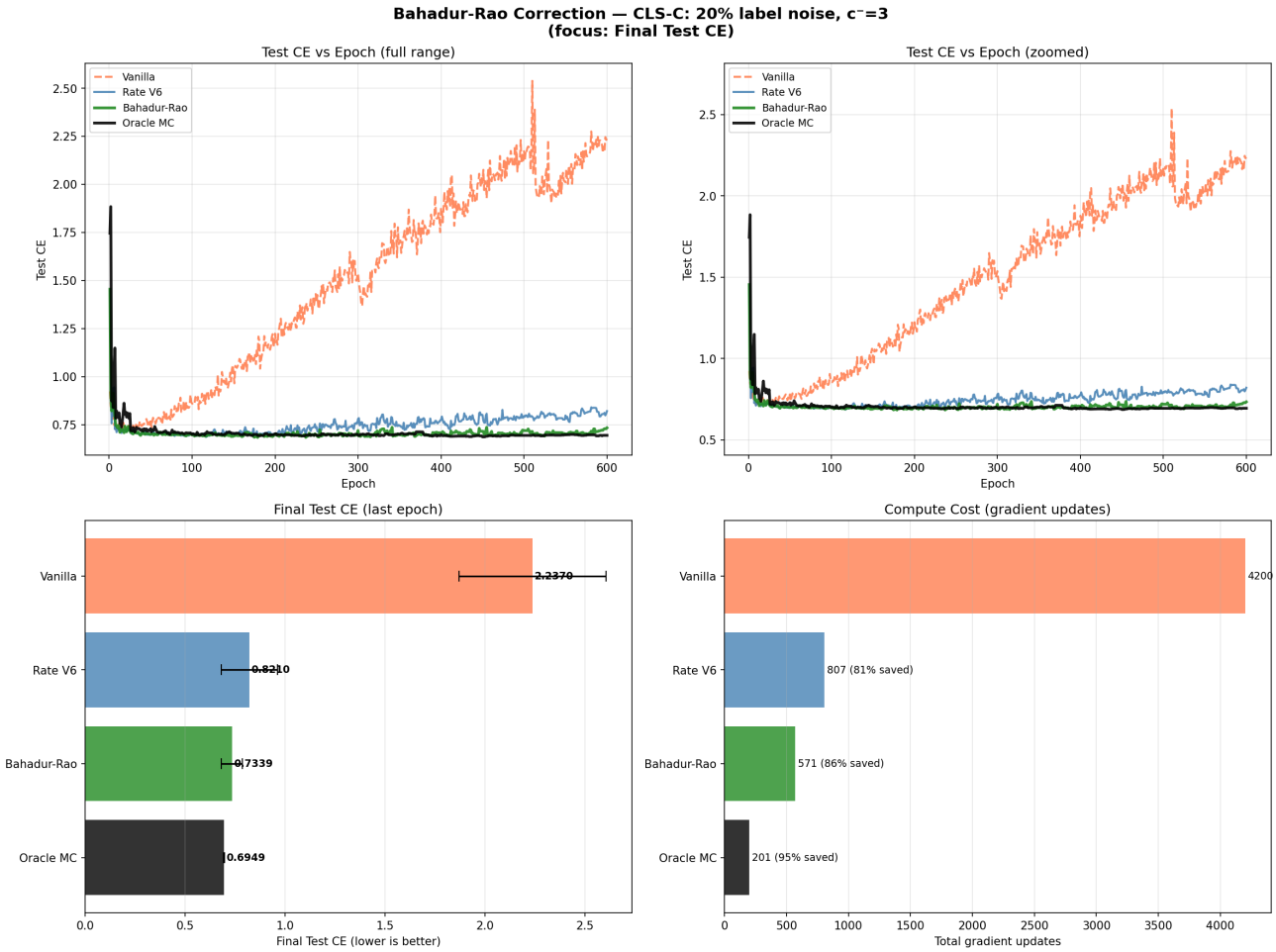

Figure 7. Saddlepoint correction on CLS-C. The same pattern holds under label noise and asymmetric correction: V6 attains 0.821, BR 0.734, oracle MC 0.695, while vanilla overfits to 2.237. BR matches the oracle to within 0.04 test CE while saving 86% of gradient updates (V6 81%, oracle 95%), confirming that closing the Cramér gap translates into measurable downstream gains, and that the heavier left-tail divergence between BR and oracle in Figure 5 does not propagate to test loss.

Discussion. The pattern is consistent across CLS-A and CLS-C. The bare rate function (V6) achieves most of the benefit relative to vanilla, and the saddlepoint correction closes the residual gap to oracle on downstream test CE, where BR matches oracle to within 0.04 test CE on both configurations, while saving 70–86% of gradient updates. The deeper consequence is visible in the real dataset experiments (CIFAR10, CIFAR100, OBGB-MOLBACE and OGBG-MOLCLINTOX). BR's saddlepoint correction pushes hazard estimates upward in both tails, producing more aggressive rejection of atypically easy and atypically hard batches alike. This delivers the tighter loss-marginal match that ECE measures, with BR achieving lower calibration error than V6 on every architecture family in Table 1. The same aggressiveness, however, discards more gradient signal on noisier tasks where the optimiser would still extract value from easy batches, and V6 retains a small accuracy/AUC edge on CIFAR-100 and on MolClinTox. The two estimators therefore occupy different points on a calibration–accuracy frontier rather than dominating one another, and the choice between them is a deployment decision: BR where calibration is the primary deliverable, V6 where task performance dominates.

D. Experiments

The CIFAR experiments were run on two Nvidia RTX 6000 Ada, each with 48GB GDDR6 ECC memory. The graph experiments were carried out on two Nvidia RTX Titan with 24GB of memory each.

D.1. Graph Classification

We conduct experiments on the OGBG-MOLCLINTOX and OGBG-MOLBACE benchmarks using standard scaffold splits. We employ a suite of five GNN backbones: GIN, GCN, GATv2, GraphSAGE, and Graph Transformer, each configured with 5 layers. Following optimised defaults from the literature, the hidden dimensionality is set to 300 for GIN, GCN, GATv2, and Graph Transformer, and 256 for GraphSAGE; multi-head architectures utilize 4 attention heads. Dropout rates are configured at 0.2 (GIN, SAGE), 0.3 (GATv2, Transformer), and 0.1 (GCN). Models are trained for 500 epochs with a batch size of 32 using SGD with momentum (0.9), a learning rate of 0.01, and a weight decay of 10^{-5}. We utilize a ReduceLROnPlateau scheduler for all variants, calibrated on the held-out validation set. For SGDR variants, the rate function approximation is carried out on the validation set. We set c^- and c^+ to 1.0 for all rejection variants. We utilize a 10-epoch warmup period before activating the rejection mechanism. For the imbalanced ClinTox dataset, we implement stratified batching and inverse-frequency class weighting to stabilize rate function estimation across all models. We compare a vanilla SGD baseline against two proposed variants: SGDR (the V6 variant chosen from the ablation studies in Section C) and the Bahadur-Rao rate function surrogate. All results are reported as the mean across three independent seeds (42, 142, 242) plus standard deviation.

D.2. Image Classification

We evaluate the proposed method on the CIFAR-10 and CIFAR-100 benchmarks across a diverse range of architectures, including VGG (11, 13, 16, and 19 with batch normalization), ResNet (20, 32, 44, and 56), ShuffleNetV2 (x0.5, x1.0, x1.5, and x2.0), and MobileNetV2 (x0.5, x0.75, and x1.0). All models are trained for 200 epochs using a standard 40k/10k/10k train/validation/test split derived from the 50k training images and 10k test images. Optimization is performed using SGD with momentum (0.9), a batch size of 128, an initial learning rate of 0.1, and weight decay of 5×10^{-4}. We utilize a cosine annealing learning rate scheduler for all variants. For SGDR variants, we implement a 10-epoch warmup period before activating the rejection mechanism. The rate function approximation is performed on the 10k-sample held-out validation set. We set c^- and c^+ to 1.0 for all rejection variants. All results are reported as the mean across three independent seeds (42, 142, 242).

The tables below report Accuracy, Brier Score, ECE, Overfit Gap (test CE - train CE) and Test Loss (CE) for each individual configuration.

D.2.1. CIFAR 10

Model Name	method	Accuracy	Brier Score	ECE	Overfit Gap	Test Loss
MobileNetv2 0.5	Baseline SGD	87.87 ± 0.95	0.1786 ± 0.0139	0.0336 ± 0.0056	0.1400 ± 0.0133	0.3687 ± 0.0291
MobileNetv2 0.5	SGDR (BR)	89.54 ± 0.53	0.1538 ± 0.0088	0.0163 ± 0.0032	0.1340 ± 0.0078	0.3130 ± 0.0183
MobileNetv2 0.5	SGDR	89.46 ± 0.81	0.1554 ± 0.0106	0.0230 ± 0.0057	0.1468 ± 0.0143	0.3209 ± 0.0233
MobileNetv2 0.75	Baseline SGD	88.74 ± 0.44	0.1670 ± 0.0064	0.0311 ± 0.0023	0.1491 ± 0.0071	0.3491 ± 0.0156
MobileNetv2 0.75	SGDR (BR)	90.55 ± 0.41	0.1392 ± 0.0055	0.0174 ± 0.0051	0.1405 ± 0.0035	0.2822 ± 0.0117
MobileNetv2 0.75	SGDR	90.64 ± 0.32	0.1382 ± 0.0035	0.0221 ± 0.0025	0.1522 ± 0.0052	0.2855 ± 0.0058
MobileNetv2 1.0	Baseline SGD	89.61 ± 0.22	0.1543 ± 0.0032	0.0289 ± 0.0021	0.1497 ± 0.0026	0.3212 ± 0.0077
MobileNetv2 1.0	SGDR (BR)	91.06 ± 0.34	0.1308 ± 0.0053	0.0151 ± 0.0029	0.1368 ± 0.0076	0.2644 ± 0.0117
MobileNetv2 1.0	SGDR	90.89 ± 0.23	0.1339 ± 0.0038	0.0206 ± 0.0015	0.1509 ± 0.0041	0.2745 ± 0.0069
ResNet20	Baseline SGD	90.23 ± 0.50	0.1506 ± 0.0073	0.0444 ± 0.0057	0.2657 ± 0.0146	0.3407 ± 0.0196
ResNet20	SGDR (BR)	90.12 ± 0.11	0.1447 ± 0.0004	0.0200 ± 0.0014	0.1691 ± 0.0040	0.2975 ± 0.0029
ResNet20	SGDR	90.39 ± 0.32	0.1414 ± 0.0033	0.0263 ± 0.0031	0.1862 ± 0.0028	0.2906 ± 0.0036
ResNet32	Baseline SGD	90.90 ± 0.38	0.1415 ± 0.0056	0.0482 ± 0.0012	0.2786 ± 0.0086	0.3289 ± 0.0179
ResNet32	SGDR (BR)	90.74 ± 0.21	0.1357 ± 0.0034	0.0219 ± 0.0007	0.1697 ± 0.0053	0.2770 ± 0.0085
ResNet32	SGDR	91.16 ± 0.29	0.1297 ± 0.0033	0.0268 ± 0.0032	0.1909 ± 0.0023	0.2708 ± 0.0054
ResNet44	Baseline SGD	91.00 ± 0.28	0.1420 ± 0.0056	0.0479 ± 0.0020	0.2960 ± 0.0157	0.3408 ± 0.0230
ResNet44	SGDR (BR)	90.88 ± 0.10	0.1322 ± 0.0007	0.0227 ± 0.0011	0.1739 ± 0.0016	0.2700 ± 0.0036
ResNet44	SGDR	91.73 ± 0.16	0.1229 ± 0.0028	0.0224 ± 0.0003	0.1838 ± 0.0041	0.2552 ± 0.0053
ResNet56	Baseline SGD	91.68 ± 0.20	0.1318 ± 0.0032	0.0459 ± 0.0023	0.2908 ± 0.0185	0.3269 ± 0.0177
ResNet56	SGDR (BR)	91.23 ± 0.29	0.1306 ± 0.0021	0.0205 ± 0.0014	0.1689 ± 0.0034	0.2651 ± 0.0035
ResNet56	SGDR	91.76 ± 0.30	0.1230 ± 0.0026	0.0237 ± 0.0025	0.1843 ± 0.0076	0.2533 ± 0.0057
ShuffleNetv2 0.5	Baseline SGD	86.75 ± 0.34	0.1954 ± 0.0049	0.0343 ± 0.0059	0.1589 ± 0.0198	0.4053 ± 0.0071
ShuffleNetv2 0.5	SGDR (BR)	87.22 ± 0.25	0.1852 ± 0.0034	0.0213 ± 0.0025	0.1483 ± 0.0044	0.3790 ± 0.0050
ShuffleNetv2 0.5	SGDR	87.07 ± 0.32	0.1864 ± 0.0060	0.0267 ± 0.0047	0.1588 ± 0.0192	0.3822 ± 0.0152
ShuffleNetv2 1.0	Baseline SGD	88.67 ± 0.38	0.1696 ± 0.0053	0.0435 ± 0.0026	0.2241 ± 0.0162	0.3661 ± 0.0133
ShuffleNetv2 1.0	SGDR (BR)	90.13 ± 0.11	0.1471 ± 0.0019	0.0216 ± 0.0033	0.1671 ± 0.0051	0.3005 ± 0.0038
ShuffleNetv2 1.0	SGDR	89.96 ± 0.16	0.1479 ± 0.0014	0.0281 ± 0.0016	0.1824 ± 0.0033	0.3024 ± 0.0049
ShuffleNetv2 1.5	Baseline SGD	89.91 ± 0.64	0.1535 ± 0.0106	0.0423 ± 0.0053	0.2442 ± 0.0177	0.3390 ± 0.0259
ShuffleNetv2 1.5	SGDR (BR)	90.78 ± 0.29	0.1352 ± 0.0066	0.0188 ± 0.0014	0.1678 ± 0.0036	0.2768 ± 0.0125
ShuffleNetv2 1.5	SGDR	90.99 ± 0.39	0.1345 ± 0.0041	0.0239 ± 0.0032	0.1845 ± 0.0070	0.2779 ± 0.0060
ShuffleNetv2 2.0	Baseline SGD	90.05 ± 0.06	0.1535 ± 0.0018	0.0460 ± 0.0019	0.2737 ± 0.0064	0.3466 ± 0.0031
ShuffleNetv2 2.0	SGDR (BR)	90.87 ± 0.20	0.1331 ± 0.0013	0.0184 ± 0.0027	0.1679 ± 0.0046	0.2714 ± 0.0031
ShuffleNetv2 2.0	SGDR	91.17 ± 0.17	0.1322 ± 0.0007	0.0242 ± 0.0026	0.1868 ± 0.0067	0.2714 ± 0.0015
VGG11	Baseline SGD	89.54 ± 0.33	0.1704 ± 0.0067	0.0645 ± 0.0026	0.3719 ± 0.0106	0.4239 ± 0.0176
VGG11	SGDR (BR)	89.38 ± 0.23	0.1566 ± 0.0011	0.0309 ± 0.0063	0.2148 ± 0.0082	0.3287 ± 0.0015
VGG11	SGDR	89.91 ± 0.39	0.1502 ± 0.0060	0.0344 ± 0.0051	0.2404 ± 0.0130	0.3225 ± 0.0125
VGG13	Baseline SGD	91.26 ± 0.46	0.1435 ± 0.0083	0.0560 ± 0.0037	0.3210 ± 0.0128	0.3620 ± 0.0219
VGG13	SGDR (BR)	91.40 ± 0.22	0.1266 ± 0.0025	0.0247 ± 0.0025	0.1912 ± 0.0078	0.2694 ± 0.0045
VGG13	SGDR	91.88 ± 0.02	0.1225 ± 0.0014	0.0306 ± 0.0021	0.2074 ± 0.0059	0.2616 ± 0.0062
VGG16	Baseline SGD	91.25 ± 0.46	0.1435 ± 0.0056	0.0564 ± 0.0021	0.3285 ± 0.0058	0.3638 ± 0.0135
VGG16	SGDR (BR)	91.43 ± 0.17	0.1282 ± 0.0025	0.0248 ± 0.0041	0.1915 ± 0.0087	0.2728 ± 0.0078
VGG16	SGDR	91.24 ± 0.19	0.1305 ± 0.0035	0.0344 ± 0.0047	0.2207 ± 0.0060	0.2833 ± 0.0071
VGG19	Baseline SGD	90.68 ± 0.26	0.1532 ± 0.0046	0.0602 ± 0.0008	0.3527 ± 0.0084	0.3954 ± 0.0124
VGG19	SGDR (BR)	91.24 ± 0.27	0.1298 ± 0.0030	0.0218 ± 0.0033	0.1892 ± 0.0094	0.2792 ± 0.0067
VGG19	SGDR	91.50 ± 0.16	0.1260 ± 0.0028	0.0270 ± 0.0034	0.2050 ± 0.0070	0.2712 ± 0.0051

D.2.2. CIFAR 100

Model Name	method	Accuracy	Brier Score	ECE	Overfit Gap	Test Loss
MobileNetv2 0.5	Baseline SGD	64.39 ± 0.90	0.4780 ± 0.0106	0.0666 ± 0.0050	0.5464 ± 0.0155	1.2716 ± 0.0339
MobileNetv2 0.5	SGDR (BR)	64.95 ± 0.29	0.4639 ± 0.0026	0.0224 ± 0.0017	0.3856 ± 0.0110	1.2098 ± 0.0094
MobileNetv2 0.5	SGDR	65.89 ± 0.39	0.4555 ± 0.0044	0.0302 ± 0.0053	0.4354 ± 0.0087	1.1888 ± 0.0088
MobileNetv2 0.75	Baseline SGD	66.26 ± 0.39	0.4578 ± 0.0061	0.0705 ± 0.0047	0.5985 ± 0.0213	1.2099 ± 0.0196
MobileNetv2 0.75	SGDR (BR)	66.72 ± 0.62	0.4442 ± 0.0043	0.0203 ± 0.0048	0.3994 ± 0.0104	1.1478 ± 0.0096
MobileNetv2 0.75	SGDR	67.48 ± 0.46	0.4364 ± 0.0063	0.0301 ± 0.0018	0.4522 ± 0.0101	1.1265 ± 0.0243
MobileNetv2 1.0	Baseline SGD	66.84 ± 0.31	0.4544 ± 0.0047	0.0763 ± 0.0033	0.6318 ± 0.0289	1.2076 ± 0.0207
MobileNetv2 1.0	SGDR (BR)	67.25 ± 0.09	0.4380 ± 0.0043	0.0244 ± 0.0035	0.4167 ± 0.0078	1.1270 ± 0.0145
MobileNetv2 1.0	SGDR	67.82 ± 0.42	0.4299 ± 0.0057	0.0328 ± 0.0048	0.4635 ± 0.0147	1.1064 ± 0.0187
resnet20	Baseline SGD	63.82 ± 0.37	0.4960 ± 0.0034	0.1019 ± 0.0023	0.7780 ± 0.0093	1.3794 ± 0.0135
resnet20	SGDR (BR)	62.57 ± 0.36	0.4930 ± 0.0051	0.0346 ± 0.0013	0.4050 ± 0.0090	1.3117 ± 0.0131
resnet20	SGDR	63.58 ± 0.37	0.4828 ± 0.0042	0.0429 ± 0.0047	0.4623 ± 0.0110	1.2832 ± 0.0106
resnet32	Baseline SGD	65.01 ± 0.77	0.4950 ± 0.0104	0.1325 ± 0.0064	1.0379 ± 0.0244	1.4366 ± 0.0225
resnet32	SGDR (BR)	63.89 ± 0.18	0.4752 ± 0.0038	0.0410 ± 0.0043	0.4247 ± 0.0110	1.2585 ± 0.0177
resnet32	SGDR	65.29 ± 0.29	0.4617 ± 0.0039	0.0442 ± 0.0033	0.4795 ± 0.0135	1.2222 ± 0.0083
resnet44	Baseline SGD	65.38 ± 1.61	0.4979 ± 0.0231	0.1500 ± 0.0103	1.1692 ± 0.0425	1.4781 ± 0.0865
resnet44	SGDR (BR)	64.83 ± 0.08	0.4676 ± 0.0032	0.0401 ± 0.0013	0.4462 ± 0.0034	1.2329 ± 0.0026
resnet44	SGDR	66.27 ± 0.28	0.4530 ± 0.0024	0.0458 ± 0.0030	0.5045 ± 0.0049	1.1940 ± 0.0037
resnet56	Baseline SGD	66.35 ± 0.35	0.4932 ± 0.0026	0.1550 ± 0.0047	1.2413 ± 0.0268	1.4964 ± 0.0121
resnet56	SGDR (BR)	65.36 ± 0.18	0.4612 ± 0.0022	0.0406 ± 0.0036	0.4480 ± 0.0059	1.2174 ± 0.0114
resnet56	SGDR	66.11 ± 0.33	0.4537 ± 0.0025	0.0504 ± 0.0049	0.5230 ± 0.0145	1.2011 ± 0.0038
shufflenetv2 0.5	Baseline SGD	62.64 ± 0.82	0.5043 ± 0.0101	0.0970 ± 0.0103	0.7990 ± 0.0225	1.3902 ± 0.0304
shufflenetv2 0.5	SGDR (BR)	61.80 ± 0.18	0.5003 ± 0.0061	0.0295 ± 0.0013	0.4098 ± 0.0162	1.3454 ± 0.0133
shufflenetv2 0.5	SGDR	62.92 ± 0.47	0.4901 ± 0.0071	0.0361 ± 0.0032	0.4701 ± 0.0149	1.3125 ± 0.0181
shufflenetv2 1.0	Baseline SGD	67.10 ± 0.61	0.4686 ± 0.0058	0.1300 ± 0.0047	1.1531 ± 0.0116	1.3367 ± 0.0049
shufflenetv2 1.0	SGDR (BR)	65.60 ± 0.86	0.4558 ± 0.0101	0.0281 ± 0.0048	0.4603 ± 0.0110	1.2014 ± 0.0316
shufflenetv2 1.0	SGDR	66.32 ± 0.85	0.4466 ± 0.0099	0.0418 ± 0.0075	0.5230 ± 0.0152	1.1764 ± 0.0278
shufflenetv2 1.5	Baseline SGD	69.03 ± 0.57	0.4453 ± 0.0108	0.1240 ± 0.0041	1.1802 ± 0.0339	1.2656 ± 0.0404
shufflenetv2 1.5	SGDR (BR)	66.01 ± 0.24	0.4513 ± 0.0014	0.0312 ± 0.0040	0.4702 ± 0.0248	1.1813 ± 0.0046
shufflenetv2 1.5	SGDR	67.43 ± 0.42	0.4351 ± 0.0030	0.0385 ± 0.0058	0.5264 ± 0.0072	1.1373 ± 0.0100
shufflenetv2 2.0	Baseline SGD	71.24 ± 0.55	0.4126 ± 0.0048	0.1089 ± 0.0036	1.1290 ± 0.0037	1.1664 ± 0.0018
shufflenetv2 2.0	SGDR (BR)	67.10 ± 0.14	0.4386 ± 0.0027	0.0310 ± 0.0047	0.4730 ± 0.0051	1.1456 ± 0.0102
shufflenetv2 2.0	SGDR	68.10 ± 0.35	0.4280 ± 0.0026	0.0396 ± 0.0025	0.5443 ± 0.0039	1.1152 ± 0.0064
VGG11	Baseline SGD	64.57 ± 0.56	0.5382 ± 0.0075	0.1960 ± 0.0055	1.7886 ± 0.0275	1.8523 ± 0.0327
VGG11	SGDR (BR)	63.24 ± 0.27	0.4851 ± 0.0020	0.0488 ± 0.0021	0.5965 ± 0.0060	1.3780 ± 0.0042
VGG11	SGDR	64.08 ± 0.62	0.4782 ± 0.0076	0.0637 ± 0.0025	0.6927 ± 0.0037	1.3698 ± 0.0166
VGG13	Baseline SGD	68.43 ± 0.45	0.4848 ± 0.0059	0.1745 ± 0.0030	1.5466 ± 0.0237	1.6120 ± 0.0298
VGG13	SGDR (BR)	66.29 ± 0.22	0.4491 ± 0.0011	0.0486 ± 0.0075	0.5820 ± 0.0050	1.2483 ± 0.0052
VGG13	SGDR	67.71 ± 0.13	0.4355 ± 0.0017	0.0624 ± 0.0045	0.6479 ± 0.0232	1.2067 ± 0.0093
VGG16	Baseline SGD	67.14 ± 0.61	0.5090 ± 0.0028	0.1942 ± 0.0020	1.5271 ± 0.0261	1.6940 ± 0.0243
VGG16	SGDR (BR)	65.02 ± 0.20	0.4615 ± 0.0020	0.0514 ± 0.0045	0.5679 ± 0.0067	1.2925 ± 0.0052
VGG16	SGDR	66.35 ± 0.15	0.4521 ± 0.0025	0.0624 ± 0.0060	0.6421 ± 0.0060	1.2739 ± 0.0020
VGG19	Baseline SGD	66.39 ± 0.70	0.5051 ± 0.0095	0.1812 ± 0.0042	1.4128 ± 0.0082	1.6683 ± 0.0386
VGG19	SGDR (BR)	62.64 ± 0.35	0.4868 ± 0.0050	0.0425 ± 0.0042	0.5713 ± 0.0130	1.3986 ± 0.0156
VGG19	SGDR	64.63 ± 0.66	0.4675 ± 0.0076	0.0529 ± 0.0087	0.6285 ± 0.0170	1.3404 ± 0.0230