Navigating the Boolean Hypercube: Sharpness-Aware Minimization Directly in Discrete Loss Landscapes

Ba-Hien Tran
Huawei Technologies France SASU
Paris Research Center
ba.hien.tran@huawei.com

Abstract

Sharpness-Aware Minimization (SAM) improves generalization in deep learning with continuous weights. However, applying it to binary neural networks through latent-weight heuristics suffers from a geometric mismatch: continuous perturbations do not faithfully probe the discrete loss landscape on $\{-1, +1\}^n$. We introduce Hamming-SAM, the first sharpness-aware optimizer that operates natively on the Boolean hypercube. It replaces the Euclidean ℓ_p-ball with a k-bit Hamming ball and solves the resulting discrete min-max problem by greedy ascent followed by sharpness-aware bit-flip descent. The objective is theoretically justified through the lens of PAC-Bayes, compression, and distributionally robust optimization. We prove an approximation guarantee for the ascent step and a finite-time convergence bound for the descent. Both are governed by the discrete interaction Hessian, the classical second-order derivative of pseudo-Boolean analysis, which serves as the unifying quantity connecting ascent quality to convergence rate. We further connect discrete flatness to Forman-Ricci curvature and algorithmic complexity. Experiments on a wide range of models and datasets demonstrate consistent improvements in in-distribution accuracy and out-of-distribution robustness over standard binary training and latent-space SAM baselines.

1 Introduction

Deep learning faces a growing tension between the computational demands of high-dimensional continuous optimization and the physical limitations of hardware at the edge. Traditional neural architectures rely on weight optimization within Euclidean space. This approach has been very successful, but at the cost of high energy consumption and large memory footprints. Interest has therefore returned to discrete models, in particular Binarized Neural Networks (BNNs) [26, 49], e.g., BitNet [55]. A parallel line of work has progressively removed the real-valued weights traditionally used to train such models. [23] showed that latent weights act as inertia rather than as parameters in their own right, and the Boolean Logic Deep Learning (BOLD) framework [42, 54] removes them altogether, operating directly on the Boolean hypercube $\{-1, +1\}^n$. However, optimizing over a discrete domain raises challenges absent from continuous training. The loss landscape is defined on an exponentially large combinatorial space (2^n configurations), gradient-based methods cannot be applied directly, and finding a global minimum is NP-hard in general [4]. The loss surface also exhibits complex interactions between bit-flips, where flipping two bits together can produce a loss change far exceeding the sum of their individual contributions. This makes the landscape very different from the smooth manifolds of continuous optimization.

Despite their efficiency, these models share a key weakness. Optimizers tend to converge toward *sharp* minima, that is, weight configurations where a few bit-flips cause large loss increases, which degrades generalization. While the connection between flat minima and generalization is well-established in continuous space [25, 28, 27], the relationship is subtle: [14] showed that common flatness measures are sensitive to reparameterization in continuous networks, complicating their use as generalization predictors. Sharpness-Aware Minimization (SAM) [19] has become an effective and well-founded

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

remedy with strong theoretical foundations [33, 29], yet transferring these insights to the binary or discrete domain remains unexplored. Existing approaches apply SAM to BNNs and quantized networks through latent-weight proxies [34, 35, 47]. Some reshape the perturbation to counter the sign barrier, but all of them perturb a continuous surrogate. This approach faces two geometric pathologies, which we formalize in § 3. Unless the perturbation is reshaped, its signal *vanishes* as latent weights grow, and in every case Euclidean proximity *fails to predict* Hamming proximity. More broadly, a theoretical understanding of flatness in discrete parameter spaces remains lacking. What flatness means there, how to measure it, and how to optimize for it are all open questions. In contrast, the reparameterization ambiguities that affect continuous flatness measures [14] are *absent* on the Boolean hypercube. Since $\{-1,+1\}^n$ admits no smooth rescaling symmetries, Hamming-ball sharpness is an unambiguous measure of the loss landscape geometry.

Contributions. This paper addresses these issues with Hamming-SAM, a native Boolean optimizer that replaces the continuous ℓ_p-ball with a k-bit Hamming ball, the natural neighborhood on $\{-1,+1\}^n$, and solves the resulting discrete min-max problem via a two-phase greedy procedure. The framework is not tied to any binary training method. Its two ingredients are the first- and second-order discrete derivatives of the loss, classical objects of pseudo-Boolean function theory [10, 43] that are defined on $\{-1,+1\}^n$ independently of how they are computed. Any per-bit discrete-derivative oracle can therefore serve as the backend, such as BOLD's chain rule [42] or a straight-through estimator (STE). Our theoretical analysis is general and does not depend on the specific backend used. Our contributions span both formulation and analysis: ❶ the objective is theoretically justified from three perspectives, namely PAC-Bayes, compression, and distributionally robust optimization (§ 4); ❷ an approximation guarantee for the greedy ascent based on the submodularity ratio, with graceful degradation for non-submodular landscapes (Theorem 5.2); ❸ a finite-time convergence proof linking the SAM-style descent to the discrete interaction Hessian (Theorems 5.7 and 5.8), which serves as the unifying quantity governing both ascent quality and convergence rate; ❹ connections between discrete flatness, Forman-Ricci curvature, and algorithmic complexity (§ 6); and ❺ experiments on CIFAR-10/100 and ImageNet with VGG, ResNet, and ViT demonstrating consistent improvements in clean accuracy and out-of-distribution robustness over binary training and latent-space SAM baselines (§ 7).

2 Related Work

SAM and its variants. [19] introduced SAM as a min-max optimizer seeking parameters with uniformly low loss in an ℓ_p-ball neighborhood. Subsequent work refined the perturbation geometry [33], reduced cost [36], and improved the ascent direction [30]. [29] recently established convergence of continuous SAM by viewing it as inexact gradient descent. All existing variants operate exclusively in continuous Euclidean parameter spaces.

Binary neural networks. Modern BNNs maintain latent real-valued weights and project them via the sign function, using the Straight-Through Estimator (STE) for gradient approximation [26, 49], with architectural advances [38, 37, 58, 56]. A separate line asks whether the latent weights are needed at all. [23] reinterpreted them as inertia and optimized the signs directly, and BOLD [42] removed them and introduced a chain rule that yields a learning signal for every bit in a single backward pass. We adopt that chain rule as our backend, but the sharpness framework developed here is independent of it and applies to any oracle for the same classical quantity. [35, 47] apply SAM to BNNs, but operate on latent continuous weights. [34] documented "perturbation diminishment" in quantized networks, partially anticipating our Vanishing Perturbation Effect.

Flat minima in discrete spaces. [1] identified dense clusters of solutions in binary networks, and [2] formalized local entropy as a discrete flatness measure. Their optimization relies on replicated SGD and belief propagation. Our objective (1) instead minimizes the worst-case loss over a Hamming ball, which yields a deterministic guarantee rather than a statistical one.

Pseudo-Boolean optimization and discrete curvature. The discrete Hessian is rooted in pseudo-Boolean function theory [10, 5], where second-order discrete derivatives have been studied extensively [21, 43]. We apply these tools to the loss landscape of Boolean networks, connecting it to the submodularity ratio that governs the ascent and to the convergence rate of the descent. Forman-Ricci curvature [20] has been applied to data graphs [53] and network architectures [50]. We are the first to apply it to a neural network's *parameter space*.

3 The Geometric Pathology of Latent-Weight SAM

The primary motivation for developing a native Boolean SAM arises from two geometric pathologies of applying continuous SAM to binarized networks (see also [34] for related observations in quantized settings). In the standard approach, high-precision latent weights $\tilde{w} \in \mathbb{R}^n$ are maintained and projected onto the Boolean space via $w = \operatorname{sign}(\tilde{w}) \in \{-1, +1\}^n$. When continuous SAM is applied to these latent weights, it searches for an adversarial perturbation ϵ within an ℓ_p-ball of radius ρ to maximize the loss. This approach suffers from two failures that we formalize below.

3.1 The Vanishing Perturbation Effect

Proposition 3.1 (Vanishing Perturbation). *Let $\mathcal{C}(\rho) = \{i : |\tilde{w}_i| > \rho\}$ be the "confident" coordinates. For any ϵ with $\|\epsilon\|_\infty \leq \rho$, $\operatorname{sign}(\tilde{w}_i + \epsilon_i) = \operatorname{sign}(\tilde{w}_i)$ for all $i \in \mathcal{C}(\rho)$. If $|\mathcal{C}(\rho)| = n$, SAM reduces to standard gradient descent on the latent weights.*

The proof is available in § C. In practice, as training progresses and latent weights grow in magnitude, the fraction of confident coordinates $|\mathcal{C}(\rho)|/n$ approaches 1, and the continuous SAM objective reduces to standard gradient descent (see Fig. 1 for an illustration).

3.2 Euclidean–Hamming Geometric Mismatch

The second failure comes from a geometric mismatch: flatness in the continuous latent space does not imply robustness in the actual discrete operating domain. Writing d_H for the Hamming distance:

Proposition 3.2 (Metric Distortion). *Map $\pi : \mathbb{R}^n \to \{-1, +1\}^n$ ($\pi(\tilde{w}) = \operatorname{sign}(\tilde{w})$) can exhibit arbitrarily large metric distortion: for any $D > 0$, there exist $\tilde{u}, \tilde{v} \in \mathbb{R}^n$ with $\|\tilde{u} - \tilde{v}\|_2 < D^{-1}$ but $d_H(\pi(\tilde{u}), \pi(\tilde{v})) = n$, and conversely, $\tilde{u}', \tilde{v}'$ with $\|\tilde{u}' - \tilde{v}'\|_2 > D$ but $d_H(\pi(\tilde{u}'), \pi(\tilde{v}')) = 0$.*

The proof is available in § C. The distortion arises because the sign function collapses continuous neighborhoods: two points straddling the origin in every coordinate are ε-close in ℓ_2 but maximally far in Hamming distance. Euclidean proximity is therefore neither necessary nor sufficient for Hamming proximity. Fig. 2 illustrates this phenomenon in $\mathbb{R}^2$.

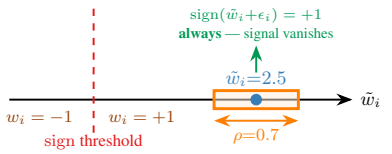


Figure 1: **The vanishing perturbation effect** (Proposition 3.1). Continuous SAM perturbs $\tilde{w}_i = 2.5$ within a ball of radius $\rho = 0.7$. Since the ball never crosses the sign threshold, the sharpness signal vanishes.

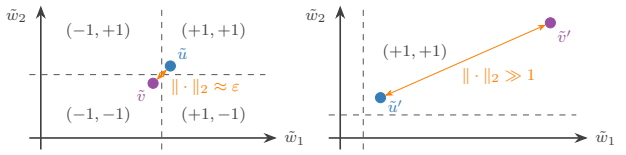


Figure 2: **Euclidean–Hamming metric distortion** (Proposition 3.2, n=2). *Left:* Two latent-weight vectors $\tilde{u}, \tilde{v}$ that are ε-close in $\mathbb{R}^2$ but straddle the sign boundary, producing $d_H = 2$. *Right:* Two vectors far apart but in the same quadrant, so $d_H = 0$. Euclidean proximity is neither necessary nor sufficient for Hamming proximity.

4 Hamming-SAM: Formulation and Algorithm

4.1 The Hamming-Ball Min-Max Objective

Let $w \in \{-1, +1\}^n$ denote the n Boolean weights in our model, $\mathcal{L}(w)$ the population loss, and $\hat{\mathcal{L}}(w) \equiv \hat{\mathcal{L}}(w; \mathcal{D})$ the empirical loss on training data $\mathcal{D}$ (we omit $\mathcal{D}$ when clear from context). Table 3 summarizes our main notations. The Hamming-SAM framework replaces the continuous ℓ_p-ball with the k-bit Hamming ball $\mathcal{B}_k(w) = \{w' \in \{-1, +1\}^n : d_H(w, w') \leq k\}$ and solves:

$$\min_{w \in \{-1,+1\}^n} \mathcal{L}_{\text{adv}}(w) := \max_{w' \in \mathcal{B}_k(w)} \hat{\mathcal{L}}(w'; \mathcal{D}). \tag{1}$$

The adversarial loss is defined as the worst-case empirical loss over all configurations within k bit-flips of the current weights. This objective is the native Boolean analogue of the PAC-Bayes generalization bound that motivates the original continuous SAM objective [19]. We give two justifications and one equivalent reading below. The detailed proofs are in § D. A structural comparison between continuous SAM and Hamming-SAM is given in Table 1.

Justification I: PAC-Bayes upper bound (Boolean analogue of [19]). In continuous SAM, [19] use a Gaussian posterior $Q = \mathcal{N}(w, \rho^2 I)$ so that the expected loss is controlled by the worst case over the ℓ_2-ball. We repeat this argument on the Boolean hypercube using a Bernoulli bit-flip distribution. Consider the distribution P centered at w that independently flips each bit with

Table 1: Continuous vs. Boolean landscape analysis.

Component	Continuous SAM	Hamming-SAM (Boolean)	Reference
Parameter space	$\mathbb{R}^n$	$\{-1,+1\}^n$	Eq. 1
Perturbation set	ℓ_2-ball $\{\epsilon : \|\epsilon\|_2 \le \rho\}$	Hamming ball $\mathcal{B}_k(w)$	Eq. 1
PAC-Bayes stochastic posterior	$\mathcal{N}(w, \rho^2 I)$	$\mathrm{BernoulliFlip}(w, q)$	Theorem 4.1
PAC-Bayes complexity (data-indep. prior)	$h(\|w\|_2^2/\rho^2)$ via Gaussian KL	$\mathrm{KL}(P\|P_0)$ via Bernoulli KL	Theorem 4.1
PAC-Bayes complexity (data-dep. prior)	$\|w - w_0\|_2^2/\sigma^2$ via Gaussian KL	$d_H(\hat{w}, w_0)\log(1/p)$ via Bernoulli KL	Lemma 5.5
First-order	Gradient $\nabla\mathcal{L}$	Boolean Variation BV_i	§ 4.2
Second-order	Hessian $\nabla^2\mathcal{L}$	Interaction Hessian H_{ij}	Eq. 7
Curvature	Hessian eigenvalues	Forman-Ricci Ric_F	Proposition K.4
Regularizer	Weight decay	BDM complexity	§ L

probability $q = k/(2n)$. Splitting the expectation over $\mathcal{B}_k(w)$ and its complement, and bounding the tail mass of the Binomial distribution, $\beta_k = \Pr[\mathrm{Bin}(n, q) > k] \le e^{-k/6}$, via Chernoff [8, 6]:

$$\mathbb{E}_{w'\sim P}[\hat{\mathcal{L}}(w')] \;\le\; (1-\beta_k)\,\mathcal{L}_{\mathrm{adv}}(w) + \beta_k B, \tag{2}$$

where $B = \max_{w'} \hat{\mathcal{L}}(w')$. Combining with the PAC-Bayes–McAllester inequality [40]:

Theorem 4.1 (Discrete PAC-Bayes Sharpness Bound)**.** *For any prior P_0 independent of the training set $\mathcal{S}$ of size m, and any $\delta \in (0,1)$, with probability $\ge 1-\delta$:*

$$\mathbb{E}_{w'\sim P}\big[\mathcal{L}(w')\big] \;\le\; \underbrace{(1-\beta_k)\,\mathcal{L}_{\mathrm{adv}}(w) + \beta_k B}_{\textit{sharpness-aware empirical term}} + \underbrace{\sqrt{\frac{\mathrm{KL}(P\|P_0) + \log(2\sqrt{m}/\delta)}{2m}}}_{\textit{complexity term}}, \tag{3}$$

where $P = \mathrm{BernoulliFlip}(w, q)$ with $q = k/(2n)$, $\beta_k \le e^{-k/6}$, and $B = \max_{w'} \hat{\mathcal{L}}(w')$.

The structure mirrors [19]: the sharpness-aware term replaces $\max_{\|\epsilon\|\le\rho} \hat{\mathcal{L}}(w+\epsilon)$, with $\beta_k \le e^{-k/6}$ replacing the Gaussian tail, while the complexity term uses a Bernoulli KL in place of a Gaussian KL. The key difference is that our bound operates directly on $\{-1,+1\}^n$. Since β_k is exponentially small for practical k, the bound is effectively $\mathbb{E}_{w'\sim P}[\mathcal{L}(w')] \le \mathcal{L}_{\mathrm{adv}}(w) + \mathrm{KL}$ term, directly motivating the minimization of $\mathcal{L}_{\mathrm{adv}}(w)$. With a uniform prior the complexity term is vacuous whenever $n > m$, and § 5.2 replaces it with a data-dependent prior. Full proof in § D.

Justification II: Compression bound. If the Hamming-SAM solution $\hat{w}$ satisfies $\mathcal{L}_{\mathrm{adv}}(\hat{w}) \le \epsilon$, *every* configuration in $\mathcal{B}_k(\hat{w})$ also has empirical loss $\le \epsilon$. A flat solution therefore covers an entire ball of low-loss configurations at once, so we can use a PAC-Bayes posterior $Q_{\hat{w}}$ that is uniform over $\mathcal{B}_k(\hat{w})$, with a uniform prior:

$$\mathbb{E}_{w'\sim Q_{\hat{w}}}\big[\mathcal{L}(w')\big] \le \epsilon + \sqrt{\frac{n\log 2 - \log V_k + \log(2\sqrt{m}/\delta)}{2m}}, \tag{4}$$

where $V_k = \sum_{j=0}^{k}\binom{n}{j}$ and $\mathrm{KL}(Q_{\hat{w}}\|P_0) = n\log 2 - \log V_k$ exactly. The bound controls the expected loss of a network drawn at random from the ball, that is, $\hat{w}$ after a random set of up to k bit flips (§ D.2). The empirical term is the worst-case loss over the ball, and the complexity term is reduced by $\log V_k \approx k\log(n/k)$, because the bound pays for the whole ball rather than for a single configuration. With a uniform prior this reduction is small relative to $n\log 2$, and the bound is vacuous when $n \gg m$. The data-dependent prior of § 5.2 yields a non-vacuous bound, in which flatness again keeps the complexity term small.

An equivalent perspective: distributionally robust optimization. The objective (1) can be written as a DRO problem [48], $\mathcal{L}_{\mathrm{adv}}(w) = \max_{\mu\in\Delta(\mathcal{B}_k(w))} \mathbb{E}_{w'\sim\mu}[\hat{\mathcal{L}}(w')]$, where $\Delta(\mathcal{B}_k(w))$ is the set of distributions supported on the Hamming ball, and the maximum is attained at the Dirac measure on the worst-case point. This places Hamming-SAM within robust optimization, with the Hamming ball in the role that a Wasserstein ambiguity set plays in the continuous setting [32] (§ D.3).

Remark 4.2 (The role of k). In all three perspectives, k controls a bias-sharpness tradeoff: *(i)* in the PAC-Bayes view, larger k makes β_k smaller (better concentration) but forces $\mathcal{L}_{\mathrm{adv}}$ higher; *(ii)* in the compression view, larger k increases $\log V_k$ (more complexity reduction) but requires uniformly low loss over a larger set; *(iii)* in the DRO view, larger k gives more conservative population coverage but a harder optimization problem.

4.2 Boolean Variation as a Discrete Gradient

The first-order discrete derivative of the loss at bit i is the finite difference

$$\mathrm{BV}_i(w) \;=\; \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w), \tag{5}$$

where $w^{(i)}$ is w with bit i negated, so it is the loss change caused by flipping the current value of bit i. It is defined on $\{-1,+1\}^n$ regardless of how it is obtained. We use it throughout the theoretical sections, where it keeps the set function, the interaction Hessian (7) and the convergence analysis free of sign factors.

Any backend that estimates this signal can be used, such as BOLD or an STE. We mainly use BOLD [42] for convenience, since it yields all n signals in a single backward pass, and confirm the gains with STE. The BOLD's chain rule (Theorem 3.12 of [42]) enables backpropagation through deep Boolean networks. The BOLD variation of the loss with respect to weight w_i is

$$f_i'(w) = \mathbf{xnor}\big(\delta(w_i \to \neg w_i),\, \delta\hat{\mathcal{L}}(w \to w^{(i)})\big), \tag{6}$$

where $\delta(\cdot \to \cdot)$ denotes the direction of change. In the $\{-1,+1\}$ conversion, where $\mathbf{xnor}$ corresponds to multiplication, this evaluates to $f_i'(w) = -w_i\,\mathrm{BV}_i(w) = \hat{\mathcal{L}}(w|_{w_i=+1}) - \hat{\mathcal{L}}(w|_{w_i=-1})$, the partial derivative of pseudo-Boolean function theory [10, 43]. It compares the two values of bit i rather than flipping it. Thus, it plays the role of a gradient: a bit is flipped when $f_i' \cdot w_i > 0$, equivalently $\mathrm{BV}_i < 0$, which moves w_i against the sign of f_i' and decreases the loss.

Exact values and first-order estimates. Evaluating Eq. 5 exactly for every bit would require $n+1$ forward passes, one per flipped configuration. A backward pass instead returns all n signals at once, as a first-order estimate: flipping bit i moves w_i by $-2w_i$, so a Taylor expansion gives $\mathrm{BV}_i(w) \approx -2w_i\,\partial\hat{\mathcal{L}}/\partial w_i$. The positive factor 2 is immaterial to Algorithm 1, since **Phase 1** selects bits by rank and **Phase 2** by sign, and both are invariant to positive rescaling. The analysis below is stated for the exact quantity Eq. 5. § G quantifies how an inexact oracle affects the ascent guarantee.

4.3 The Hamming-SAM Algorithm

Algorithm 1 illustrates our optimization procedure (example shown in Fig. 3). **Phase 1** greedily identifies the k bits that most increase the loss, constructing an adversarial point in $\mathcal{B}_k(w^t)$. **Phase 2** evaluates the Boolean Variation at this *worst-case* point and flips the single bit that most reduces the loss, updating the original weights rather than the adversarial ones. Computing descent at this worst-case point ensures robustness against sharp loss landscapes. Using a backward pass (e.g., BOLD, § B), each "Compute BV_i" loop requires just one backward pass. The total per-step cost is therefore two backward passes (at w^t and $\hat{w}$) (see details in § P).

Algorithm 1: Hamming-SAM Optimization Step

Input: Current weights $w^t \in \{-1,+1\}^n$, Hamming radius k

Phase 1: Greedy Ascent (Adversarial Probing)
for *each bit* $i = 1,\dots,n$ **do** // Done by single backward pass
  Compute $\mathrm{BV}_i(w^t)$;
$\mathcal{I} \leftarrow$ argtop-k$\{\mathrm{BV}_i(w^t) : \mathrm{BV}_i(w^t) > 0\}$ // $\le k$ indices with largest positive BV
$\hat{w} \leftarrow w^t$ with bits in $\mathcal{I}$ flipped // Adversarial point $\hat{w} \in \mathcal{B}_k(w^t)$

Phase 2: Sharpness-Aware Descent (Weight Update)
for *each bit* $i = 1,\dots,n$ **do** // Done by single backward pass
  Compute $\mathrm{BV}_i(\hat{w})$;
$i^* \leftarrow \arg\min_{i \notin \mathcal{I}} \mathrm{BV}_i(\hat{w})$ // Exclude ascent bits
if $\mathrm{BV}_{i^*}(\hat{w}) < 0$ **then**
  $w^{t+1} \leftarrow w^t$ with bit i^* flipped;
else
  $w^{t+1} \leftarrow w^t$;

Remark 4.3 (Practical Phase 2). With the BOLD backend, **Phase 2** feeds the backward signal at the adversarial point, an estimate of $f_i'(\hat{w}^t)$, into the accumulator of [42] for every bit, which flips bit i when the accumulated evidence for $f_i' \cdot w_i^t > 0$ exceeds an adaptive threshold (§ I). For $i \notin \mathcal{I}$, $f_i'(\hat{w}^t)\, w_i^t = -\mathrm{BV}_i(\hat{w}^t)$, which is positive exactly when the descent test $\mathrm{BV}_i(\hat{w}^t) < 0$ of Algorithm 1 holds. For an ascent bit, the signal predicts that restoring it lowers the loss, so the step moves its accumulator away from the flip threshold (§ I).

Remark 4.4 (Connection to local entropy). The Hamming-SAM objective Eq. 1 is related to but distinct from the local entropy of [2], which maximizes $\log|\{w' \in \mathcal{B}_k(w) : \mathcal{L}(w') \le \theta\}|$, with a threshold θ. Hamming-SAM minimizes the worst-case loss (a min-max objective), while local entropy maximizes the volume of low-loss configurations (a counting objective). Both encourage convergence to dense clusters of good solutions, but Hamming-SAM provides a deterministic, worst-case guarantee.

4.4 Choosing the Hamming Radius k

k is the central hyperparameter, controlling the adversarial probe strength and the generalization bound tightness. The compression bound Eq. 4 decomposes into a sharpness term $\mathcal{L}_{\mathrm{adv}}(\hat{w}; k)$,

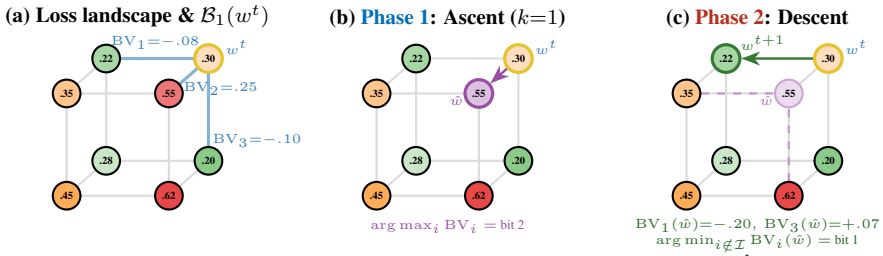


Figure 3: **Hamming-SAM walkthrough on $\{-1,+1\}^3$ with k=1.** Node values show loss $\hat{\mathcal{L}}$ (green = low, red = high). **(a)** Current weight w^t=(+1, +1, +1) (loss 0.30); blue edges show $\mathcal{B}_1(w^t)$ and Boolean Variations. **(b) Phase 1** selects bit 2 (BV=0.25, the only increase) as the adversarial point $\hat{w}$=(+1, −1, +1). **(c) Phase 2** evaluates BV at $\hat{w}$ and flips bit 1 ($\mathrm{BV}_1(\hat{w})$=−0.20), yielding w^{t+1}=(−1, +1, +1) with loss 0.22. Base-point descent would instead flip bit 3, reaching a point whose one-bit worst case is 0.62 rather than 0.35.

increasing in k, plus a complexity term, decreasing in k with reduction $\log V_k \approx k \log(n/k)$ for $k \ll n$.

***Definition* 4.5** ((L,σ)-Smooth Boolean Landscape)**.** *The loss is (L,σ)-smooth if $\mathcal{L}_{\mathrm{adv}}(w;k) - \hat{\mathcal{L}}(w) \le L \cdot k^\sigma$ for all w and $k \ge 1$, where $\sigma > 0$ controls how fast the worst-case loss grows with the radius: $\sigma < 1$ indicates diminishing returns, and $\sigma > 1$ that each extra flip adds more than the previous one.*

Proposition 4.6 (Optimal k)**.** *Under (L,σ)-smoothness with $k \ll n$ and $\sigma > 1$, the compression bound is approximately minimized at $k^* \approx \left(\log n \,/\, (2L\sigma\sqrt{2mn\log 2})\right)^{1/(\sigma-1)}$.*

The case $\sigma > 1$ arises when flipping bits together costs more than the sum of their individual effects. For $\sigma \le 1$ the bound has no interior minimum and is minimized at the boundary. The full statement for all σ and its proof are in § E. Since $\log V_k \approx n\,\alpha \log(1/\alpha)$ for $k = \alpha n$, a fixed fraction α removes the same share of the description length at every model size, which motivates parametrizing the radius as $k = \lfloor \alpha \cdot n \rfloor$ across model sizes. We use $k = 500$ in most cases ($\alpha \approx 10^{-5}$ to 10^{-3}), and the ablation of § 7.2 shows a broad plateau in k.

5 Theoretical Analysis

5.1 Approximation Guarantee for the Ascent Step

A key requirement is that the greedy ascent identifies a near-worst-case configuration within $\mathcal{B}_k(w)$. In continuous SAM, the ascent is justified as a first-order Taylor approximation. In the discrete domain, finding the adversarial point $\hat{w}$ is a cardinality-constrained maximization of the non-negative risk-increase set function $f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$ for $|S| \le k$, where $w^t \oplus \mathbf{1}_S$ denotes flipping the bits indexed by the set S. The quality of the greedy search depends on the approximate submodularity of this function. We use the framework of [3]:

***Definition* 5.1** (Submodularity Ratio and Curvature [11, 3])**.** *Let $\rho_j(A) = f(A \cup \{j\}) - f(A)$. The submodularity ratio $\gamma_k \in (0,1]$ is the largest γ such that $\sum_{j \in L} \rho_j(S) \ge \gamma\,[f(S \cup L) - f(S)]$ for all disjoint L, S with $|L| \le k$. The generalized curvature $\alpha \in [0,1]$ is the smallest scalar such that $\rho_i\big((S \setminus \{i\}) \cup \Omega\big) \ge (1-\alpha)\,\rho_i(S \setminus \{i\})$ for all S, Ω and $i \in S \setminus \Omega$.*

The submodularity ratio quantifies how close f is to submodular ($\gamma = 1$ for submodular functions). The curvature α measures how much the marginal gain of an element can shrink when other elements are added: $\alpha = 0$ for modular functions, and $\alpha = 1$ allows gains to vanish entirely (the submodular worst case, recovering the classical $(1 - 1/e)$ bound) [41].

Theorem 5.2 (Sequential Greedy Ascent Guarantee)**.** *Let $\mathcal{V}_+ = \{i \in [n] : \mathrm{BV}_i(w^t) > 0\}$ and suppose f, restricted to subsets of $\mathcal{V}_+$, is non-negative, non-decreasing (monotone), and has submodularity ratio γ_k and curvature α (Definition 5.1). Let $\hat{S}$ be the set of k indices selected by the sequential greedy (which re-evaluates marginal gains after each selection) and $S^* = \arg\max_{S \subseteq \mathcal{V}_+, |S| \le k} f(S)$. Then: $f(\hat{S}) \ge \frac{1}{\alpha}\left[1 - e^{-\gamma_k \alpha}\right] f(S^*)$. When γ_k=1, α=1 (submodular), this recovers the classical $(1-1/e)$ bound.*

The full proof is available in § G.

Remark 5.3 (Monotonicity). Every bit in $\mathcal{V}_+$ increases the loss when flipped alone. Flipping other bits first can shrink this gain through pairwise interactions, so f is monotone on $\mathcal{V}_+$ as long as each

gain exceeds what interactions can remove. § G makes this precise in terms of the interaction Hessian introduced below.

Connection to the discrete interaction Hessian. The quality of the greedy approximation depends on the landscape curvature, characterized by the discrete interaction Hessian, the second-order flip difference of pseudo-Boolean function theory [10, 43]:

$$H_{ij}(w) = \hat{\mathcal{L}}(w^{(ij)}) - \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w^{(j)}) + \hat{\mathcal{L}}(w). \tag{7}$$

This quantity measures the pairwise interaction between bit-flips i and j. If $H_{ij} = 0$, the two bits contribute independently to the loss change. If $H_{ij} < 0$, flipping both gives *less* increase than the sum of individual flips (diminishing returns), and if $H_{ij} > 0$, it gives *more* (the flips reinforce each other). See § F for more details.

The connection to the ascent guarantee is direct. The second-order discrete derivative of f at context S satisfies $\Delta_{ij} f(S) = H_{ij}(w^t \oplus \mathbf{1}_S)$ (§ G). Therefore, f is submodular, in which case $\gamma_k = 1$, if and only if $H_{ij}(w^t \oplus \mathbf{1}_S) \leq 0$ for all S and $i, j \notin S$. When the Hessian has positive entries, the submodularity ratio γ_k decreases below 1 and the ascent guarantee degrades gracefully: the bound $(1/\alpha)[1 - e^{-\gamma_k \alpha}]$ is continuous and positive for any $\gamma_k > 0$.

The one-shot rule used in practice. Algorithm 1 selects the k bits in a single backward pass rather than re-evaluating marginal gains k times. Expanding f in the Möbius basis of pseudo-Boolean analysis isolates the cost of this choice: the top-k rule already maximizes the first-order term exactly, so the deficit is carried entirely by pairwise and higher-order interactions.

Theorem 5.4 (Top-k Ascent Guarantee)**.** *Let $\hat{S}_{\text{tk}}$ be the top-k selection of Algorithm 1, restricted to bits with $\text{BV}_i > 0$. Suppose $|H_{ij}(w)| \leq h$ for all w and $i \neq j$, and the third- and higher-order interaction residual satisfies $|R(S)| \leq R_{\max}$ for all $|S| \leq k$. Then $f(\hat{S}_{\text{tk}}) \geq f(S^*) - k(k-1)h - 2R_{\max}$.*

Proof and empirical instantiation are in § G.

Roles of the two guarantees. Theorems 5.2 and 5.4 are complementary rather than nested, and both are needed. They constrain the same object through different features of it. By the equivalence above, submodularity is a condition on the *sign* structure of the interaction Hessian, whereas Theorem 5.4 depends only on its *magnitude*. Neither assumption set implies the other. Theorem 5.2 is multiplicative and scale-free, certifying a constant fraction of the optimum however large $f(S^*)$ is, but it requires monotonicity and degrades as $\gamma_k \to 0$. Theorem 5.4 does not require monotonicity or submodularity, but is informative only while $k(k-1)h$ stays small relative to $f(S^*)$. The ascent is therefore near-optimal whenever *either* the landscape is approximately submodular *or* its pairwise interactions are weak. § N.1 reports sampled submodularity ratios close to one throughout training, consistent with the first condition. The two also differ in cost. The sequential greedy is the strongest ascent available with unlimited compute, at k backward passes per step, while Algorithm 1 runs the one-shot rule in a single pass. The gap between them, measured in § N.4, is the price of that tractability.

5.2 Generalization via Data-Dependent Priors

The PAC-Bayes justification in § 4 (Theorem 4.1) uses a uniform prior, yielding a KL divergence of $n \log 2$ that is vacuous for any model with $n > m$. Tighter bounds are possible with data-dependent Bernoulli priors [17]: split $\mathcal{D} = \mathcal{S}_0 \sqcup \mathcal{S}_1$, train a reference w_0 on $\mathcal{S}_0$, and define the prior as a product distribution that flips each bit of w_0 independently with probability p.

Lemma 5.5 (KL for Bernoulli Bit-Flip Prior)**.** *For the deterministic posterior $Q = \delta_{\hat{w}}$ and prior P with flip probability p: $\text{KL}(Q\|P) = d \log(1/p) + (n-d)\log(1/(1-p))$, where $d = d_H(\hat{w}, w_0)$.*

The resulting McAllester-style bound scales as $\mathcal{O}(d \log(1/p))$ rather than $\mathcal{O}(n \log 2)$, and is non-vacuous when $d \ll n$. The advantage of Boolean weights is that this KL is *exact and closed-form*. Hamming-SAM fits this framework well. Flat basins are *stable* under the distribution shift from $\mathcal{S}_0$ to $\mathcal{S}_1$, requiring fewer bit-flips to adapt and keeping d small, whereas sharp minima must escape the basin and incur a large Hamming distance. Lower sharpness B_k thus yields smaller adaptation distance d and a tighter bound. Full derivation in § J.

5.3 Convergence Analysis

We present two convergence results for the empirical loss $\hat{\mathcal{L}}(w^t)$. [29] recently established convergence of continuous SAM under diminishing stepsizes and Lipschitz gradients. Our discrete setting

differs in that *(i)* $\hat{\mathcal{L}}(w^t)$ decreases *monotonically* at each accepted step, and *(ii)* the convergence condition involves the discrete Hessian rather than Lipschitz constants. Remark 5.9 additionally bounds the adversarial objective $\mathcal{L}_{\text{adv}}(w^t)$ along the trajectory.

***Definition* 5.6** (η-Discrete Stationarity)**.** *A configuration w is η-stationary if $\text{BV}_i(w) \geq -\eta$ for all i, so that no single bit-flip can reduce $\hat{\mathcal{L}}(w)$ by more than η.*

Theorem 5.7 (Finite-Time Convergence (Base-Point Descent))**.** *Suppose $\hat{\mathcal{L}} : \{-1,+1\}^n \to [0,B]$. Consider a variant of Algorithm 1 where Phase 2 evaluates BV_i at w^t (rather than at $\hat{w}$), selecting $i^* = \arg\min_i \text{BV}_i(w^t)$ and flipping only when $\text{BV}_{i^*}(w^t) < -\eta$. Then this variant converges to an η-stationary point in at most $T \leq \lfloor B/\eta \rfloor$ iterations.*

The proof is in § H. We now show that the full SAM-style algorithm also converges, provided the discrete Hessian is bounded.

Theorem 5.8 (Finite-Time Convergence (SAM-Style Descent))**.** *Suppose $\hat{\mathcal{L}} : \{-1,+1\}^n \to [0,B]$ and $|H_{ij}(w)| \leq h$ for all w and $i \neq j$. In Algorithm 1, suppose Phase 2 selects $i^* \notin \mathcal{I}$ and accepts the flip only when $\text{BV}_{i^*}(\hat{w}) < -\eta$ with $\eta > k \cdot h$. Then each accepted step satisfies $\hat{\mathcal{L}}(w^{t+1}) \leq \hat{\mathcal{L}}(w^t) - (\eta - kh)$, and convergence occurs in at most $T \leq \lfloor B/(\eta - kh) \rfloor$ iterations.*

The key idea is a telescoping identity: since $\hat{w}$ and w^t differ in the k ascent bits $\mathcal{I}$, restoring these bits one at a time yields $\text{BV}_{i^*}(w^t) = \text{BV}_{i^*}(\hat{w}) + \sum_{l=1}^{k} H_{i^*,j_l}(u^{l-1})$, where each $|H_{i^*,j_l}| \leq h$. Full proof in § H.

Remark 5.9 (Controlling the adversarial objective). The expansion behind Theorem 5.4 also bounds the sharpness itself: $\mathcal{L}_{\text{adv}}(w) - \hat{\mathcal{L}}(w) \leq k\,B_1(w) + \binom{k}{2}h + R_{\max}$, where $B_1(w) = \max_i \max(\text{BV}_i(w), 0)$ is the one-bit sharpness, available from a single backward pass. Since Theorem 5.8 drives $\hat{\mathcal{L}}(w^t)$ down monotonically, $\mathcal{L}_{\text{adv}}(w^t)$ stays within an explicit and observable band above it, and full min-max convergence reduces to controlling B_1, which Phase 2 targets directly. At $k = 1$ the bound is an identity.

Remark 5.10 (The condition $\eta > kh$ as a design principle). This condition admits a natural algorithmic interpretation. The descent signal must exceed the total interaction between i^* and the k ascent bits, and the practitioner enforces this by choosing k small (default $k/n \approx 10^{-4}$) or η large. Structurally, h is moderate because bits in different neurons of the same layer perturb disjoint pre-activations and can interact only through downstream layers. With threshold activations a single flip shifts a pre-activation by exactly ± 2, so whenever either margin absorbs the shift, $H_{ij} = 0$ exactly. With the real-valued activations we train, the interaction is nonzero but collapses during training (§ 7.2). The condition also reinforces itself. The algorithm tends to converge to low-h regions, which have high γ_k (Theorem 5.2), so the discrete Hessian is the *unifying quantity* governing both ascent quality and convergence rate.

Multi-bit descent and the accumulator. In practice, an accumulator such as BOLD's may flip m bits simultaneously. The following proposition (proof in § I) quantifies when this is safe:

Proposition 5.11 (Multi-Bit Descent under Bounded Hessian)**.** *Under $|H_{ij}(w)| \leq h$, if m bits are flipped simultaneously, each with $\text{BV}_{i_l}(w^t) < -\delta$ and $\delta > (m-1)h/2$, then $\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t) \leq -m\delta + \binom{m}{2}h < 0$.*

Combined with Theorem 5.8, a multi-bit step decreases the loss when $\eta > (k + \frac{m-1}{2})h$, a three-way tradeoff between probe strength (k), optimizer aggressiveness (m), and landscape curvature (h). The condition applies when every flipped bit satisfies the premise of Proposition 5.11. The accumulator used in practice selects bits by accumulated signal, and § I analyzes its effective threshold.

6 Diagnostic Extensions

Two further constructions are natural on the hypercube. We develop them in the appendix as a diagnostic and as an optional extension, not as part of the method or of its guarantees.

Forman-Ricci curvature. Weighting the Hamming graph by the loss and by the Boolean Variations yields a Forman-Ricci curvature [20] on the parameter space. Curvature is high when the loss and the Boolean Variations vary little across the edges incident to a configuration, which is the combinatorial

signature of a flat basin. Reducing the sharpness $B_k = \mathcal{L}_{\text{adv}}(w) - \hat{\mathcal{L}}(w)$ raises a lower bound on its average (§ K). It thus provides a computable descriptive statistic of local flatness (§ 7.2).

Algorithmic-complexity regularization. Because Hamming-SAM proceeds by bit-flips rather than gradients, a non-differentiable complexity measure such as the Block Decomposition Method (BDM) [57] can enter the objective directly, at $\mathcal{O}(1)$ cost per candidate flip and with a convergence bound of the same form. [51] observed that BDM decreases during BNN training but could not use it as a training-time regularizer, since it is non-differentiable. Hamming-SAM removes this obstacle, making algorithmic-complexity regularization available as an optional extension of the method (§ L).

7 Experiments

We evaluate Hamming-SAM on ResNet-18 [22], VGG-16 [52], and ViT [15] on CIFAR-10/100 [31] and their corrupted variants [24], as well as ResNet-50 on ImageNet [13]. All binary methods replace full-precision layers with Boolean (BOLD) or binary (STE) counterparts without modifying the architecture or adding learnable parameters. We compare: full-precision (FP), STE [26], STE with latent-space SAM [19] (STE-SAM), standard BOLD, and Hamming-SAM on the BOLD backend (k=500 by default, with a k-ablation in § 7.2). Details are in the Appendix, with additional theoretical verifications in § N.

7.1 Generalization and Robustness

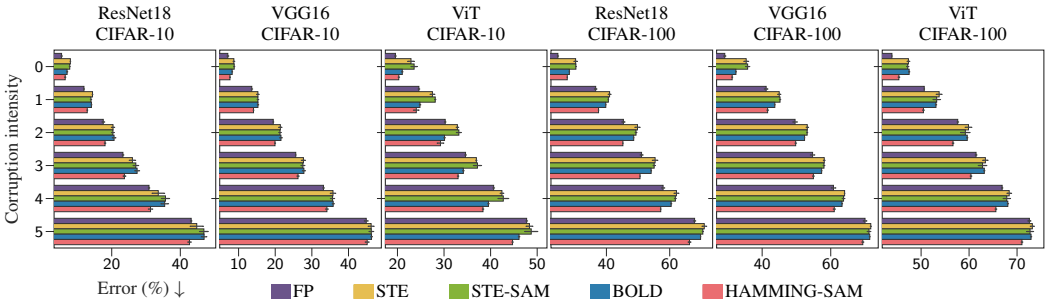


Figure 4: Error rate (%, ↓) at corruption intensities for different architectures on CIFAR-10 and CIFAR-100.

Fig. 4 reports error rates at corruption intensities 0 (clean) through 5 across all architecture–dataset combinations. Three patterns appear consistently.

Hamming-SAM improves over all Boolean baselines. On clean CIFAR-10, Hamming-SAM reduces the error of standard BOLD across architectures. On CIFAR-100 the gains are comparable. STE-SAM, which applies continuous SAM to latent weights, provides smaller and less consistent improvements over STE, confirming that the vanishing perturbation effect (§ 3) limits the effect of latent-space sharpness-aware training.

Robustness gains exceed clean accuracy gains. As corruption intensity increases, the gap between Hamming-SAM and BOLD widens. The improvement in mean corruption accuracy exceeds the clean accuracy improvement in every setting, by a factor of 3× on average, consistent with the theoretical prediction that flat minima (low B_k) transfer better under distribution shift. On several corruption types (e.g., ResNet-18 on CIFAR-10, see Fig. 5), Hamming-SAM narrows the gap to the full-precision baseline substantially, despite using only 1-bit weights. The gains also hold without BOLD. On a standard STE backend, Hamming-SAM improves both clean accuracy and robustness over STE-SAM at the same cost (§ M.2).

Per-corruption analysis. Fig. 5 breaks down the CIFAR-10-C results for ResNet-18. Hamming-SAM reduces error on all 19 corruption types, with the largest gains on high-frequency perturbations (Glass Blur, Shot Noise, Impulse Noise) and smaller but consistent improvements on low-frequency corruptions (Brightness, Fog). This consistency suggests that sharpness reduction is a general robustness mechanism rather than a corruption-specific effect.

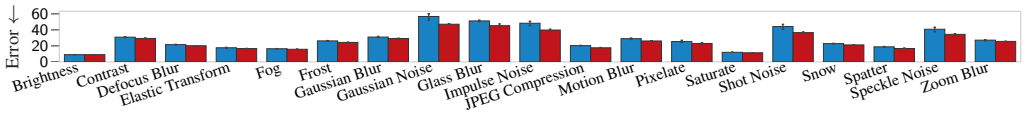
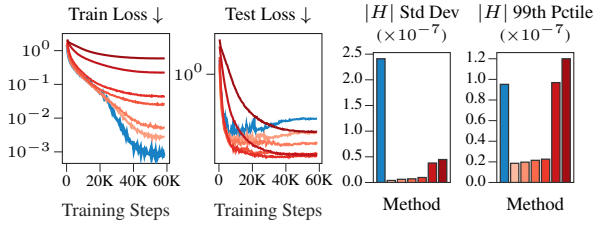


Figure 5: Average per-corruption error on CIFAR-10-C (ResNet-18). BOLD (blue) vs. Hamming-SAM (red).

7.2 Boolean Landscape Analysis and Theoretical Validation



Method	Test Acc (%) ↑	mCA (%) ↑	$\|H\|_1/\|H\|_5$ Ratio	Norm. Ric_F
BOLD	91.65 ± 0.12	70.56 ± 0.18	53.53	-2.12
HAM.-SAM ($k = 50$)	91.84 ± 0.10	71.59 ± 0.15	2.20	-2.00
HAM.-SAM ($k = 100$)	**92.31** ± 0.11	72.18 ± 0.14	2.08	-1.98
HAM.-SAM ($k = 500$)	92.15 ± 0.13	**72.39** ± 0.16	4.85	-1.96
HAM.-SAM ($k = 1000$)	91.73 ± 0.15	72.01 ± 0.19	6.06	-1.92
HAM.-SAM ($k = 5000$)	91.05 ± 0.21	70.97 ± 0.22	5.38	-1.80
HAM.-SAM ($k = 10000$)	88.31 ± 0.25	69.80 ± 0.28	1.40	-1.47

Figure 6: VGG on CIFAR-10. *Left:* Training and test loss curves. *Center:* Discrete Hessian statistics at convergence. *Right:* Test accuracy, corruption robustness (mCA), Hessian spikiness ratio, and normalized Forman-Ricci curvature across methods and Hamming radii k. Results averaged over 4 seeds (± std shown).

We use the VGG-16 / CIFAR-10 setup to validate the theoretical predictions, ablating over k (Fig. 6).

k-ablation. Test accuracy and corruption robustness both peak at moderate $k \in [100, 500]$ and degrade for large k (e.g., k=10000 falls below the BOLD baseline). This matches the bias–sharpness tradeoff of § 4.4. Moderate k probes meaningful perturbations, while excessive k forces the optimizer to defend against unrealistically large Hamming balls. Seed variance also grows with k, as the interaction term kh of Theorem 5.8 approaches the acceptance threshold and the tail of the selected set becomes less stable across minibatches.

Discrete Hessian spectrum. The standard deviation of sampled $|H_{ij}|$ drops by $\sim$40$\times$ from BOLD to Hamming-SAM k=100. The spikiness ratio $|H|_1/|H|_5$, the largest sampled $|H_{ij}|$ divided by the fifth largest, drops from 53.5 (BOLD) to between 1.4 and 6.1 across all Hamming-SAM radii. This indicates a shift from a few dominant interactions to a nearly uniform spectrum. Over 300 random bit pairs, the largest interaction entry falls by 12$\times$ during training to about 10^{-7} at convergence (§ N.2), so the interaction term kh in Theorem 5.8 remains small. Small interactions also make the loss change of a set of flips nearly additive, which keeps the submodularity ratio of Theorem 5.2 close to 1, as confirmed empirically in § N.1.

Forman-Ricci curvature. Normalized Ric_F (§ K.5) increases (becomes less negative) at Hamming-SAM solutions. This is the expected direction. Curvature is high when the loss and the Boolean Variations vary little across the edges incident to a configuration, and reducing the sharpness B_k raises a lower bound on the average curvature (Theorem K.5). The curvature is highest at k=10000, which has the flattest local neighborhood but the worst test accuracy. This reflects the trade-off in k: accuracy peaks at moderate radii while curvature keeps increasing, so curvature is a local diagnostic of flatness rather than an independent predictor of generalization.

Training dynamics. The loss curves (Fig. 6, left) illustrate the generalization mechanism: Hamming-SAM maintains higher train loss throughout but achieves monotonically decreasing test loss, while standard BOLD stalls at a worse minimum after $\sim$20K steps.

7.3 Large-scale Evaluation on ImageNet

Table 2 reports ResNet-50 results on ImageNet. Hamming-SAM improves over standard BOLD on both test accuracy and OOD AUROC (evaluated on ImageNet-C [24]), with gains of 0.69 points in top-1 accuracy and 2.65 points in OOD AUROC, consistent with the pattern on CIFAR. In contrast, STE-SAM degrades both metrics relative to STE, confirming that latent-space sharpness-aware training does not scale reliably and that the geometric pathologies of § 3 grow stronger at ImageNet scale.

Table 2: ImageNet Results.

Method	Test Acc	OOD AUC
STE	68.97 ± 0.09	77.79 ± 0.13
STE-SAM	68.53 ± 0.15	77.12 ± 0.18
BOLD	69.03 ± 0.11	78.22 ± 0.08
Hamming-SAM	**69.72** ± 0.14	**80.87** ± 0.11

8 Conclusion

We introduced Hamming-SAM, the first sharpness-aware optimizer that operates natively on the Boolean hypercube, eliminating the geometric pathologies of latent-space approaches. The theoretical framework is unified by the discrete interaction Hessian, which governs the quality of the adversarial ascent, the price of the one-shot approximation used in practice, and the convergence rate of the descent. Experiments across a wide range of architectures and datasets demonstrate consistent improvements in both clean accuracy and out-of-distribution robustness. We believe the Hamming-ball formulation and the interaction-Hessian analysis developed here may find broader applications wherever optimization over binary parameter spaces arises.

Acknowledgements and Disclosures

The author would like to thank the anonymous reviewers for their constructive comments, as well as Jean-Claude Belfiore, Li Xu, Van Minh Nguyen, Li Rong, and Yun Yaw Chu for their assistance and support. The author is an employee of Huawei Technologies France S.A.S.U., which supported this work. The author declares no other competing interests.

References

[1] C. Baldassi, A. Ingrosso, C. Lucibello, L. Saglietti, and R. Zecchina. Subdominant Dense Clusters Allow for Simple Learning and High Computational Performance in Neural Networks with Discrete Synapses. *Physical Review Letters*, 115(12):128101, 2015.

[2] C. Baldassi, F. Pittorino, and R. Zecchina. Shaping the Learning Landscape in Neural Networks around Wide Flat Minima. *Proceedings of the National Academy of Sciences*, 117(1):161–170, 2020.

[3] A. A. Bian, J. M. Buhmann, A. Krause, and S. Tschiatschek. Guarantees for Greedy Maximization of Non-submodular Functions with Applications. In *International Conference on Machine Learning (ICML)*, 2017.

[4] A. Blum and R. Rivest. Training a 3-node Neural Network is NP-complete. In *Advances in Neural Information Processing Systems (NIPS)*, volume 1, 1988.

[5] E. Boros and P. L. Hammer. Pseudo-Boolean Optimization. *Discrete Applied Mathematics*, 123:155–225, 2002.

[6] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 02 2013.

[7] O. Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*, volume 56 of *Lecture Notes–Monograph Series*. Institute of Mathematical Statistics, 2007.

[8] H. Chernoff. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations. *The Annals of Mathematical Statistics*, pages 493–507, 1952.

[9] F. R. K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

[10] Y. Crama and P. L. Hammer. *Boolean Functions: Theory, Algorithms, and Applications*, volume 142 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2011.

[11] A. Das and D. Kempe. Submodular Meets Spectral: Greedy Algorithms for Subset Selection, Sparse Approximation and Dictionary Selection. In *International Conference on Machine Learning (ICML)*, 2011.

[12] A. Das and D. Kempe. Approximate Submodularity and its Applications: Subset Selection, Sparse Approximation and Dictionary Selection. *Journal of Machine Learning Research*, 19(3):1–34, 2018.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[14] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp Minima Can Generalize For Deep Nets. In *International Conference on Machine Learning (ICML)*, 2017.

[15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[16] G. K. Dziugaite and D. M. Roy. Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In *Uncertainty in Artificial Intelligence (UAI)*, 2017.

[17] G. K. Dziugaite, K. Hsu, W. Gharbieh, G. Arpino, and D. Roy. On the Role of Data in PAC-Bayes Bounds. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2021.

[18] W. Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, New York, 3rd edition, 1968. ISBN 978-0-471-25708-0.

[19] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. Sharpness-Aware Minimization for Efficiently Improving Generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

[20] R. Forman. Bochner's Method for Cell Complexes and Combinatorial Ricci Curvature. *Discrete & Computational Geometry*, 29(3):323–374, 2003.

[21] M. Grabisch and M. Roubens. An Axiomatic Approach to the Concept of Interaction among Players in Cooperative Games. *International Journal of Game Theory*, 28(4):547–565, 1999.

[22] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] K. Helwegen, J. Widdicombe, L. Geiger, Z. Liu, K.-T. Cheng, and R. Nusselder. Latent Weights Do Not Exist: Rethinking Binarized Neural Network Optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

[24] D. Hendrycks and T. Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

[25] S. Hochreiter and J. Schmidhuber. Flat Minima. *Neural Computation*, 9(1):1–42, 1997.

[26] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio. Binarized Neural Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[27] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio. Fantastic Generalization Measures and Where to Find Them. In *International Conference on Learning Representations (ICLR)*, 2020.

[28] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. In *International Conference on Learning Representations (ICLR)*, 2017.

[29] P. D. Khanh, H.-C. Luong, B. S. Mordukhovich, and D. B. Tran. Fundamental Convergence Analysis of Sharpness-Aware Minimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[30] M. Kim, D. Li, S. X. Hu, and T. Hospedales. Fisher SAM: Information Geometry and Sharpness Aware Minimisation. In *International Conference on Machine Learning (ICML)*, 2022.

[31] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

[32] D. Kuhn, P. M. Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh. Wasserstein Distributionally Robust Optimization: Theory and Applications in Machine Learning. In *Operations Research & Management Science in the Age of Analytics*, pages 130–166. 2019.

[33] J. Kwon, J. Kim, H. Park, and I. K. Choi. ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks. In *International Conference on Machine Learning (ICML)*, 2021.

[34] J. Liu, J. Cai, and B. Zhuang. Sharpness-aware Quantization for Deep Neural Networks. *arXiv preprint arXiv:2111.12273*, 2021.

[35] R. Liu, F. Bian, and X. Zhang. Binary Quantized Network Training with Sharpness-Aware Minimization. *Journal of Scientific Computing*, 94(1):16, 2023.

[36] Y. Liu, S. Mai, X. Chen, C.-J. Hsieh, and Y. You. Towards Efficient and Scalable Sharpness-Aware Minimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[37] Z. Liu, B. Wu, W. Luo, X. Yang, W. Liu, and K.-T. Cheng. Bi-Real Net: Enhancing the Performance of 1-bit CNNs With Improved Representational Capability and Advanced Training Algorithm. In *European Conference on Computer Vision (ECCV)*, 2018.

[38] Z. Liu, Z. Shen, M. Savvides, and K.-T. Cheng. ReActNet: Towards Precise Binary Neural Network with Generalized Activation Functions. In *European Conference on Computer Vision (ECCV)*, 2020.

[39] H. Luo, T. Truong, T. Pham, M. Harandi, D. Phung, and T. Le. Explicit Eigenvalue Regularization Improves Sharpness-Aware Minimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[40] D. A. McAllester. PAC-Bayesian Stochastic Model Selection. *Machine Learning*, 51:5–21, 2003.

[41] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An Analysis of Approximations for Maximizing Submodular Set Functions—I. *Mathematical Programming*, 14(1):265–294, 1978.

[42] V. M. Nguyen, C. Ocampo, A. Askri, L. Leconte, and B.-H. Tran. BOLD: Boolean Logic Deep Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[43] R. O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

[44] Y. Ollivier. Ricci Curvature of Markov Chains on Metric Spaces. *Journal of Functional Analysis*, 256(3): 810–864, 2009.

[45] E. Parrado-Hernández, A. Ambroladze, J. Shawe-Taylor, and S. Sun. PAC-Bayes Bounds with Data-Dependent Priors. *Journal of Machine Learning Research*, 13:3507–3531, 2012.

[46] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[47] H. Pu, D. Zhang, K. Xu, R. Mo, Z. Yan, and D. Wang. BNN-SAM: Improving Generalization of Binary Object Detector by Seeking Flat Minima. *Applied Intelligence*, 54(8):6682–6700, 2024.

[48] H. Rahimian and S. Mehrotra. Frameworks and Results in Distributionally Robust Optimization. *Open Journal of Mathematical Optimization*, 3:1–85, 2022.

[49] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[50] B. Rieck, C. Bock, and K. Borgwardt. A Persistent Weisfeiler–Lehman Procedure for Graph Classification. In *International Conference on Machine Learning (ICML)*, 2019.

[51] E. Y. Sakabe, F. S. Abrahão, A. Simões, E. Colombini, P. Costa, R. Gudwin, and H. Zenil. Binarized Neural Networks Converge Toward Algorithmic Simplicity: Empirical Support for the Learning-as-Compression Hypothesis. *arXiv preprint arXiv:2505.20646*, 2025.

[52] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[53] J. Topping, F. Di Giovanni, B. P. Chamberlain, X. Dong, and M. M. Bronstein. Understanding Over-Squashing and Bottlenecks on Graphs via Curvature. In *International Conference on Learning Representations (ICLR)*, 2022.

[54] B.-H. Tran and V. M. Nguyen. Highly Efficient and Effective LLMs with Multi-Boolean Architectures. In *International Conference on Learning Representations (ICLR)*, 2026.

[55] H. Wang, S. Ma, L. Ma, L. Wang, W. Wang, L. Dong, S. Huang, H. Wang, J. Xue, R. Wang, Y. Wu, and F. Wei. BitNet: 1-bit Pre-training for Large Language Models. *Journal of Machine Learning Research*, 26 (125):1–29, 2025.

[56] X. Xing, Y. Li, W. Li, W. Ding, Y. Jiang, Y. Wang, J. Shao, C. Liu, and X. Liu. Towards Accurate Binary Neural Networks via Modeling Contextual Dependencies. In *European Conference on Computer Vision (ECCV)*, 2022.

[57] H. Zenil, S. Hernandez-Orozco, N. A. Kiani, F. Soler-Toscano, A. Rueda-Toicen, and J. Tegnér. A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity. *Entropy*, 20(8):605, 2018.

[58] Y. Zhang, Z. Zhang, and L. Lew. PokeBNN: A Binary Pursuit of Lightweight Accuracy. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Appendix

Table of Contents

A Notations **16**

B Preliminaries: The BOLD Framework **17**
- B.1 Boolean Neuron . . . 17
- B.2 Boolean Variation and Chain Rule . . . 17
- B.3 Accumulator-Based Optimizer . . . 17
- B.4 Notation Bridge: Unsigned Loss Change . . . 17

C Proofs for Pathological Behavior of Latent-Weight SAM **18**
- C.1 The Vanishing Perturbation Effect . . . 18
- C.2 Euclidean–Hamming Geometric Mismatch . . . 19

D Justifications for the Objective **19**
- D.1 Derivation I: PAC-Bayes Upper Bound . . . 19
- D.2 Derivation II: Compression-Based Generalization . . . 21
- D.3 An Equivalent Perspective: Distributionally Robust Optimization . . . 21

E Choosing the Hamming Radius k: Full Analysis **22**
- E.1 The Bias–Sharpness Decomposition . . . 22
- E.2 Optimal k from the Compression Bound . . . 23
- E.3 Volume of the Hamming Ball . . . 24
- E.4 Relationship to Continuous SAM's ρ . . . 24
- E.5 Practical Guidelines . . . 25

F Details on the Connection to the Discrete Interaction Hessian **25**

G Detailed Proof of Approximation Guarantees for the Ascent Step **27**
- G.1 Guarantee for the One-Shot Top-k Ascent . . . 29

H Details of Convergence Analysis **31**

I The BOLD Accumulator as an Adaptive Threshold **33**
- I.1 Accumulator Dynamics . . . 33
- I.2 Derivation of the Effective Threshold . . . 33
- I.3 Adaptive Behavior Across Training Phases . . . 34
- I.4 Formal Convergence Guarantee . . . 35

J Detailed Proof of Generalization Bound with Data-Dependent Priors **35**
- J.1 KL Divergence Computation . . . 36
- J.2 Generalization Bound . . . 36
- J.3 Empirical Evaluation . . . 37

K Forman-Ricci Curvature: Full Analysis **38**

K.1 The Loss-Weighted Hamming Graph . . . 38
K.2 Forman-Ricci Curvature on Weighted Graphs . . . 38
K.3 Curvature and Sharpness . . . 39
K.4 Average Curvature and Sharpness . . . 40
K.5 Estimating the Average Curvature . . . 41
K.6 Worked Example: Curvature on $\{-1,+1\}^3$ . . . 41
K.7 Discussion . . . 41

L BDM Regularization: Full Analysis and Empirical Evidence 42
L.1 The Block Decomposition Method . . . 42
L.2 BDM-Regularized Hamming-SAM Objective . . . 42
L.3 Convergence of BDM-Regularized Hamming-SAM . . . 43
L.4 Connection to the PAC-Bayes Framework . . . 43
L.5 Empirical Evaluation . . . 44

M Additional Results 46
M.1 Tabular Version of Benchmark Results . . . 46
M.2 Hamming-SAM on an STE Backend . . . 47
M.3 Visualizing the Loss Landscape . . . 48

N Empirical Verification of Theoretical Predictions 49
N.1 Submodularity Ratio γ_k (Theorem 5.2) . . . 49
N.2 Discrete Hessian Spectrum (Theorem 5.8) . . . 49
N.3 Multi-Bit Descent (Proposition 5.11) . . . 51
N.4 Sequential vs. Parallel Greedy Ascent Quality . . . 51

O Limitations and Future Work 52

P Experimental Details 52

Q Illustrative Example 53

A Notations

The main notation used throughout the paper is summarized in Table 3.

Table 3: Summary of main notations.

Symbol	Description
Setup	
n	Number of Boolean parameters
m	Number of training samples ($\lvert\mathcal{D}\rvert$ or $\lvert\mathcal{S}_1\rvert$)
$\mathcal{D}$	Training dataset
$w \in \{-1,+1\}^n$	Boolean weight vector
$w^{(i)}$	w with bit i negated
$w^{(ij)}$	w with bits i and j negated
$w^t \oplus \mathbf{1}_S$	w^t with all bits in set S flipped
Loss functions	
$\mathcal{L}(w)$	Population (true) loss: $\mathbb{E}_{(x,y)\sim\mathcal{P}}[\ell(f_w(x),y)]$
$\hat{\mathcal{L}}(w)$	Empirical loss: $\frac{1}{m}\sum_{i=1}^m \ell(f_w(x_i),y_i)$
$\mathcal{L}_{\text{adv}}(w)$	Adversarial loss: $\max_{w'\in\mathcal{B}_k(w)} \hat{\mathcal{L}}(w')$
B	Global loss bound: $\max_{w'} \hat{\mathcal{L}}(w')$
B_k	Sharpness: $\mathcal{L}_{\text{adv}}(w) - \hat{\mathcal{L}}(w)$
Hamming geometry	
$d_H(w,w')$	Hamming distance between w and w'
$\mathcal{B}_k(w)$	Hamming ball: $\{w' : d_H(w,w') \le k\}$
V_k	Hamming ball volume: $\sum_{j=0}^k \binom{n}{j}$
k	Hamming radius (adversarial probe strength)
Discrete calculus	
$f'_i(w)$	BOLD variation: $-w_i[\hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)] = \hat{\mathcal{L}}(w\lvert_{w_i=+1}) - \hat{\mathcal{L}}(w\lvert_{w_i=-1})$
$\text{BV}_i(w)$	Boolean Variation (unsigned): $\hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)$
$H_{ij}(w)$	Interaction Hessian: $\hat{\mathcal{L}}(w^{(ij)}) - \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w^{(j)}) + \hat{\mathcal{L}}(w)$
h	Hessian bound: $\max_{w,\, i\neq j} \lvert H_{ij}(w)\rvert$
Ascent and convergence	
$f(S)$	Set function: $\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$
γ_k	Submodularity ratio ($=1$ for submodular f)
α	Generalized curvature ($=0$ for modular f)
η	Descent threshold (flip when $\text{BV}_{i^*} < -\eta$)
PAC-Bayes	
$\mathcal{S}_0, \mathcal{S}_1$	Data split for prior / posterior training
w_0	Prior center (trained on $\mathcal{S}_0$)
d	Adaptation distance: $d_H(\hat{w}, w_0)$
p	Bernoulli flip probability in the prior
β_k	Tail mass outside $\mathcal{B}_k(w)$ ($\le e^{-k/6}$)
δ	Confidence parameter (bound holds w.p. $\ge 1-\delta$)
Geometric and information-theoretic perspectives	
Ric_F	Forman-Ricci curvature on the Hamming graph
β	Temperature in the loss-weighted Hamming graph
$\text{BDM}(w)$	Block Decomposition Method (algorithmic complexity)
λ	BDM regularization coefficient

B Preliminaries: The BOLD Framework

We briefly review the Boolean Logic Deep Learning (BOLD) framework [42], which we use as the backend for Hamming-SAM. BOLD removes real-valued latent weights entirely. Network parameters live natively in $\{-1,+1\}^n$ (equivalently $\{\mathrm{T},\mathrm{F}\}^n$ in Boolean logic).

B.1 Boolean Neuron

A BOLD neuron with input size m, weights $w_0, w_1, \ldots, w_m \in \{-1,+1\}$, and Boolean or real inputs computes the pre-activation:

$$s = w_0 + \textstyle\sum_{i=1}^{m} \mathrm{L}(w_i, x_i), \tag{8}$$

where L is a Boolean logic gate (typically XNOR) and the summation counts the number of TRUE outputs. The framework is flexible, as it allows Boolean linear layers to be connected through activation layers, normalization layers, or other types of layers.

B.2 Boolean Variation and Chain Rule

The BOLD variation [42] of $\hat{\mathcal{L}}$ with respect to w_i is

$$f'_i(w) = \mathbf{xnor}\big(\delta(w_i \to \neg w_i),\, \delta\hat{\mathcal{L}}(w \to w^{(i)})\big), \tag{9}$$

where $\delta(\cdot \to \cdot)$ denotes the direction of change and $w^{(i)}$ is w with bit i negated. For a real-valued loss it evaluates to

$$f'_i(w) = -w_i\big[\hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)\big] = \hat{\mathcal{L}}(w|_{w_i=+1}) - \hat{\mathcal{L}}(w|_{w_i=-1}) \tag{10}$$

(§ B.4), the partial derivative of pseudo-Boolean function theory [10, 43]. It compares the two values of bit i and does not depend on the current value of w_i. It plays the role of the gradient. We have $f'_i > 0$ when $\hat{\mathcal{L}}$ increases with w_i, and flipping bit i decreases the loss when f'_i and w_i have the same sign.

BOLD also provides a chain rule for compositions (Theorem 3.12 of [42]), under the assumptions stated there, which propagates variations through a deep Boolean network and returns a signal for all n weights in a single backward pass. Evaluating BV exactly would instead require $n+1$ forward passes. We treat the backward signal as an estimate. Flipping bit i moves w_i by $-2w_i$, so a first-order expansion gives $\mathrm{BV}_i(w) \approx -2w_i\,\partial\hat{\mathcal{L}}/\partial w_i$. For the multilinear extension of $\hat{\mathcal{L}}$ this relation is exact, since its partial derivative with respect to w_i equals f'_i divided by 2. The backward pass differentiates the network's own real-valued computation instead, so the estimate is exact only when the two agree. The positive factor 2 does not affect the algorithm, because Phase 1 ranks bits and Phase 2 tests signs. § G quantifies how an oracle of this kind affects the ascent guarantee.

B.3 Accumulator-Based Optimizer

The per-sample atomic variation $q_{i,k} = \mathrm{xnor}(\delta\mathcal{L}/\delta x_k^{l+1}, x_k^l)$ (Eq. 5 of [42]) is aggregated over the batch into an integer signal q_i (Eq. 7 of [42]). The simplest update rule flips w_i when $\mathrm{xnor}(q_i, w_i) = \mathrm{T}$, that is, when the aggregated signal indicates that flipping decreases the loss.

In practice, BOLD uses an *accumulator* a_i^t that integrates signals across iterations:

$$a_i^{t+1} = r^t a_i^t + \eta^t q_i^{t+1}, \tag{11}$$

where η^t is a learning rate and $r^t = N^t_{\mathrm{unchanged}}/N_{\mathrm{total}}$ is a *plasticity ratio* that down-weights the accumulator when many bits are flipping (high plasticity) and preserves it when the network is stable (low plasticity). A bit is flipped when $\mathbf{xnor}(a_i, w_i) = \mathrm{T}$, that is, when $a_i w_i > 0$. The reference implementation uses the thresholded form $a_i w_i \geq 1$ (Algorithm 8 of [42]), which we also use, and the accumulator resets to zero on flip. This provides an implicit, adaptive convergence threshold (analyzed in § I).

B.4 Notation Bridge: Unsigned Loss Change

For the theoretical analysis we work with the loss change of a single flip,

$$\mathrm{BV}_i(w) = \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w), \tag{12}$$

because it is the change caused by the move the algorithm makes. It removes sign factors from the set function $f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$, the interaction Hessian and the convergence analysis.

In the variation of the loss $\hat{\mathcal{L}}$ defined by BOLD, $f'_i(w) = \mathbf{xnor}\big(\delta(w_i \to \neg w_i),\ \delta\hat{\mathcal{L}}(w \to w^{(i)})\big)$, the second argument is numeric and equals the finite difference $\hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)$. With the embedding $e(\mathrm{T}) = +1$, $e(\mathrm{F}) = -1$ and $\mathbf{xnor}(a, x) = e(a)\,x$ for a logic value a and a number x (Proposition A.3 of [42]),

$$f'_i(w) = e\big(\delta(w_i \to \neg w_i)\big) \cdot \big[\hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)\big]. \tag{13}$$

Now $e(\delta(w_i \to \neg w_i)) = -w_i$. If $w_i = +1$, the direction is down, $\delta = \mathrm{F}$ and $e = -1$. If $w_i = -1$, the direction is up, $\delta = \mathrm{T}$ and $e = +1$. Therefore

$$f'_i(w) = -w_i\,\mathrm{BV}_i(w) = \hat{\mathcal{L}}(w|_{w_i=+1}) - \hat{\mathcal{L}}(w|_{w_i=-1}), \qquad \mathrm{BV}_i(w) = -w_i\,f'_i(w). \tag{14}$$

The variation compares the two values of bit i, while BV_i flips its current value. If $w_i = +1$, the flip goes from $+1$ to -1 and $\mathrm{BV}_i = -f'_i$, and if $w_i = -1$ it goes from -1 to $+1$ and $\mathrm{BV}_i = f'_i$. The product $f'_i \cdot w_i = -\mathrm{BV}_i$ is positive exactly when flipping bit i decreases the loss, while positive BV_i means that flipping it increases the loss. Since $w_i^2 = 1$, either quantity recovers the other. At second order the same sign factors give $H_{ij}(w) = w_i w_j \big[f'_i(w|_{w_j=+1}) - f'_i(w|_{w_j=-1})\big]$. In Algorithm 1, Phase 2 reduces to BOLD's flip rule in the f'_i convention. A bit is flipped when $\mathbf{xnor}(f'_i, w_i) = \mathrm{T}$, equivalently when $\mathrm{BV}_i < 0$.

C Proofs for Pathological Behavior of Latent-Weight SAM

The main motivation for a native Boolean SAM comes from the geometric pathologies of training binarized networks via latent-weight approximations. In the standard approach, high-precision latent weights $\tilde{w} \in \mathbb{R}^n$ are maintained and projected onto the Boolean space through a non-differentiable sign function, $w = \mathrm{sign}(\tilde{w}) \in \{-1, +1\}^n$ [26]. Gradients are approximated using the STE. When continuous SAM is applied to these latent weights, it searches for an adversarial perturbation ϵ within an ℓ_p-norm ball of radius ρ to maximize the loss:

$$\max_{\|\epsilon\|_p \le \rho} \mathcal{L}(\tilde{w} + \epsilon). \tag{15}$$

This approach has two failures.

C.1 The Vanishing Perturbation Effect

The sign function $\mathrm{sign}(\cdot)$ acts as a quantization barrier. For a latent weight $\tilde{w}_i$ that has been optimized to high magnitude, and therefore represents high confidence, a perturbation ϵ bounded by ρ is frequently insufficient to cross the zero threshold. Consequently, the binarized network's functional output remains identical: $\mathrm{sign}(\tilde{w}_i + \epsilon_i) = \mathrm{sign}(\tilde{w}_i)$ for all $|\epsilon_i| \le \rho < |\tilde{w}_i|$. The adversarial loss then equals the clean loss. The sharpness-aware signal vanishes, and the optimizer reverts to standard SGD. This is a binary-specific case of the perturbation shrinkage reported by [34] for multi-bit quantized networks.

Proposition C.1 (Vanishing Perturbation). *Let $\tilde{w} \in \mathbb{R}^n$ be latent weights and $w = \mathrm{sign}(\tilde{w})$. Define $\mathcal{C}(\rho) = \{i \in [n] : |\tilde{w}_i| > \rho\}$ as the set of "confident" coordinates. Then for any ϵ with $\|\epsilon\|_\infty \le \rho$, the binarized adversarial point satisfies $\mathrm{sign}(\tilde{w}_i + \epsilon_i) = \mathrm{sign}(\tilde{w}_i)$ for all $i \in \mathcal{C}(\rho)$. In particular, if $|\mathcal{C}(\rho)| = n$ (all weights confident), then $\max_{\|\epsilon\|_\infty \le \rho} \mathcal{L}(\mathrm{sign}(\tilde{w} + \epsilon)) = \mathcal{L}(\mathrm{sign}(\tilde{w}))$, and SAM reduces to standard gradient descent on the latent loss.*

Proof. If $|\tilde{w}_i| > \rho \ge |\epsilon_i|$, then $\tilde{w}_i + \epsilon_i$ has the same sign as $\tilde{w}_i$. The second claim follows by noting that the binarized output is identical for all perturbations in the ℓ_∞-ball. □

The statement is given for the ℓ_∞-ball, which is the weakest form. Since $\|\epsilon\|_2 \le \rho$ implies $\|\epsilon\|_\infty \le \rho$, the same conclusion holds for the ℓ_2-ball used by continuous SAM in practice.

C.2 Euclidean–Hamming Geometric Mismatch

The second failure comes from a geometric mismatch. Flatness in the continuous latent space does not imply robustness in the actual discrete operating domain. Two latent configurations that are close in Euclidean distance may map to Boolean configurations that are far apart in Hamming distance, and vice versa.

Proposition C.2 (Metric Distortion). *The map $\pi : \mathbb{R}^n \to \{-1, +1\}^n$ defined by $\pi(\tilde{w}) = \operatorname{sign}(\tilde{w})$ can exhibit arbitrarily large metric distortion: for any $D > 0$, there exist $\tilde{u}, \tilde{v} \in \mathbb{R}^n$ with $\|\tilde{u} - \tilde{v}\|_2 < D^{-1}$ but $d_H(\pi(\tilde{u}), \pi(\tilde{v})) = n$, and conversely, $\tilde{u}', \tilde{v}'$ with $\|\tilde{u}' - \tilde{v}'\|_2 > D$ but $d_H(\pi(\tilde{u}'), \pi(\tilde{v}')) = 0$.*

Proof. For the first claim, take $\tilde{u}_i = 1/(4nD)$ and $\tilde{v}_i = -1/(4nD)$ for all i. Then $\|\tilde{u} - \tilde{v}\|_2 = 1/(2D\sqrt{n}) < D^{-1}$ for every $n \geq 1$, while every coordinate of $\tilde{u}$ is positive and every coordinate of $\tilde{v}$ is negative, so $d_H(\pi(\tilde{u}), \pi(\tilde{v})) = n$. For the second, take $\tilde{u}'_i = 1$ and $\tilde{v}'_i = 1 + 2D/\sqrt{n}$. Then $\|\tilde{u}' - \tilde{v}'\|_2 = 2D > D$, and both vectors lie in the positive orthant, so $\pi(\tilde{u}') = \pi(\tilde{v}')$. □

Both constructions place the two points on opposite sides of the sign boundary, or strictly inside the same orthant, which is what makes the distortion unbounded in either direction. Euclidean proximity is therefore neither necessary nor sufficient for Hamming proximity.

D Justifications for the Objective

Structural parallel with [19]. The foundational theorem of continuous SAM [19] establishes that, for any $\rho > 0$, with high probability over $\mathcal{S} \sim \mathcal{D}^m$:

$$\mathcal{L}_{\mathcal{D}}(w) \;\leq\; \underbrace{\max_{\|\epsilon\|_2 \leq \rho} \hat{\mathcal{L}}_S(w + \epsilon)}_{\text{sharpness-aware loss}} + h\big(\|w\|_2^2/\rho^2\big), \tag{16}$$

where h is a strictly increasing function incorporating the KL divergence between a Gaussian posterior $Q = \mathcal{N}(w, \rho^2 I)$ and a grid of Gaussian priors. The continuous SAM objective $\min_w \max_{\|\epsilon\|_2 \leq \rho} \hat{\mathcal{L}}_S(w + \epsilon)$ is then motivated directly by minimizing the right-hand side of (16).

Hamming-SAM's Eq. (1) follows the same derivation on the Boolean hypercube, with the replacements below.

Component	Foret et al. (continuous)	Hamming-SAM (Boolean)
Parameter space	$\mathbb{R}^n$	$\{-1, +1\}^n$
Perturbation set	ℓ_2-ball $\{\epsilon : \|\epsilon\|_2 \leq \rho\}$	Hamming ball $\mathcal{B}_k(w)$
Stochastic posterior	$Q = \mathcal{N}(w, \rho^2 I)$	$Q = \text{BernoulliFlip}(w, q)$
Complexity term	$h(\|w\|_2^2/\rho^2)$ via Gaussian KL	$d_H(\hat{w}, w_0) \cdot \log(1/p)$ via Bernoulli KL

We now make this analogy precise. We give two theoretical justifications, a PAC-Bayes upper bound (the direct Boolean counterpart of [19]) and a compression-based argument, followed by an equivalent perspective of the objective as distributionally robust optimization.

D.1 Derivation I: PAC-Bayes Upper Bound

In continuous SAM, [19] use a Gaussian posterior $Q = \mathcal{N}(w, \rho^2 I)$ so that $\mathbb{E}_{\epsilon \sim Q}[\hat{\mathcal{L}}(w + \epsilon)]$ is controlled by $\max_{\|\epsilon\|_2 \leq \rho} \hat{\mathcal{L}}(w + \epsilon)$ via tail concentration of the χ^2 distribution. We repeat this argument on the Boolean hypercube using a Bernoulli bit-flip distribution as the stochastic posterior.

Consider the bit-flip distribution P centered at w that independently flips each bit with probability $q \in (0, \frac{1}{2})$. For any $w' \in \{-1, +1\}^n$ with $d_H(w', w) = j$, the distribution assigns mass $P(w') = q^j (1 - q)^{n-j}$. The expected empirical loss under P is:

$$\mathbb{E}_{w' \sim P}\big[\hat{\mathcal{L}}(w')\big] = \sum_{j=0}^{n} \sum_{w' : d_H(w', w) = j} q^j (1 - q)^{n-j}\, \hat{\mathcal{L}}(w').$$

Proposition D.1 (Adversarial Loss as a Stochastic Upper Bound). *For any $w \in \{-1,+1\}^n$ and any $q \in (0, \frac{1}{2})$:*

$$\mathbb{E}_{w' \sim P}\big[\hat{\mathcal{L}}(w')\big] \;\le\; \underbrace{\Big[1 - \sum_{j=0}^{k} \tbinom{n}{j} q^j (1-q)^{n-j}\Big]}_{\triangleq\ \beta_k} \cdot B \;+\; \big(1-\beta_k\big) \cdot \mathcal{L}_{\mathrm{adv}}(w), \tag{17}$$

where $B = \max_{w'} \hat{\mathcal{L}}(w')$ is the global loss bound and β_k is the probability mass outside $\mathcal{B}_k(w)$. In particular, for $q \le k/(2n)$, we have $\beta_k \le e^{-k/6}$ (by a Chernoff bound), so the expected loss is dominated by $\mathcal{L}_{\mathrm{adv}}(w)$.

Proof. Split the expectation over the Hamming ball and its complement:

$$\mathbb{E}_{w' \sim P}[\hat{\mathcal{L}}(w')] = \underbrace{\sum_{\substack{w': \\ d_H(w',w) \le k}} P(w')\, \hat{\mathcal{L}}(w')}_{\le\ (1-\beta_k) \cdot \mathcal{L}_{\mathrm{adv}}(w)} + \underbrace{\sum_{\substack{w': \\ d_H(w',w) > k}} P(w')\, \hat{\mathcal{L}}(w')}_{\le\ \beta_k \cdot B}. \tag{18}$$

The first inequality holds because $\hat{\mathcal{L}}(w') \le \mathcal{L}_{\mathrm{adv}}(w)$ for all $w' \in \mathcal{B}_k(w)$ and the total mass on $\mathcal{B}_k(w)$ is $1 - \beta_k$. The second holds since $\hat{\mathcal{L}}(w') \le B$ everywhere.

For the tail bound, the number of bit-flips under P is $\mathrm{Bin}(n, q)$ with mean $\mu = nq = k/2$ when $q = k/(2n)$. The standard multiplicative Chernoff bound [6] at $\delta = 1$ (i.e., $\Pr[X > 2\mu]$) gives

$$\beta_k = \Pr[\mathrm{Bin}(n,q) > k] \le \left(\frac{e}{4}\right)^{\mu} = \left(\frac{e}{4}\right)^{k/2} = e^{-\frac{\ln 4 - 1}{2} k} \le e^{-k/6},$$

since $(\ln 4 - 1)/2 \approx 0.193 \ge 1/6$. Smaller q only decreases the tail mass, so the bound holds for all $q \le k/(2n)$. □

Combining this with the standard PAC-Bayes theorem gives the Boolean analogue of the generalization bound of [19].

Theorem 4.1 (Discrete PAC-Bayes Sharpness Bound, Boolean Analogue of [19]). *Let P_0 be a prior over $\{-1,+1\}^n$ independent of the training set $\mathcal{S}$ of size m, and let P be the Bernoulli bit-flip distribution centered at w with flip probability $q = k/(2n)$. For any $\delta \in (0,1)$, with probability at least $1-\delta$ over $\mathcal{S} \sim \mathcal{D}^m$:*

$$\mathbb{E}_{w' \sim P}\big[\mathcal{L}(w')\big] \;\le\; \underbrace{(1-\beta_k) \cdot \mathcal{L}_{\mathrm{adv}}(w) + \beta_k B}_{\textit{sharpness-aware empirical term}} \;+\; \underbrace{\sqrt{\frac{\mathrm{KL}(P\|P_0) + \log(2\sqrt{m}/\delta)}{2m}}}_{\textit{complexity term}}, \tag{19}$$

where $\beta_k \le e^{-k/6}$ and the complexity term depends on the choice of prior P_0.

Proof. By the PAC-Bayes–McAllester inequality [40], for any posterior Q and data-independent prior P_0:

$$\mathbb{E}_{w' \sim Q}[\mathcal{L}(w')] \;\le\; \mathbb{E}_{w' \sim Q}[\hat{\mathcal{L}}(w')] \;+\; \sqrt{\frac{\mathrm{KL}(Q\|P_0) + \log(2\sqrt{m}/\delta)}{2m}}.$$

Set $Q = P$ (the Bernoulli bit-flip distribution centered at w). By Proposition D.1, the expected empirical loss satisfies $\mathbb{E}_{w' \sim P}[\hat{\mathcal{L}}(w')] \le (1-\beta_k)\, \mathcal{L}_{\mathrm{adv}}(w) + \beta_k B$. Substituting yields (19). □

Remark D.3 (One-to-one correspondence with [19]). The structure of Theorem 4.1 mirrors [19] exactly. *(i)* The sharpness-aware term $(1-\beta_k)\mathcal{L}_{\mathrm{adv}}(w) + \beta_k B$ plays the role of $\max_{\|\epsilon\|_2 \le \rho} \hat{\mathcal{L}}(w+\epsilon)$, with $\beta_k \le e^{-k/6}$ replacing the Gaussian tail. *(ii)* The complexity term is driven by a Bernoulli KL in place of a Gaussian KL. *(iii)* Both bounds control the *expected perturbed loss* $\mathbb{E}_{w' \sim Q}[\mathcal{L}(w')]$ rather than the deterministic loss at the center, so minimizing $\mathcal{L}_{\mathrm{adv}}(w)$ is motivated by its dominance on the right-hand side in both frameworks. The key difference is that our bound operates directly on $\{-1,+1\}^n$.

D.2 Derivation II: Compression-Based Generalization

A second justification comes from a volume argument on the Boolean hypercube. The main idea is that if w has low adversarial loss, then *every* configuration in $\mathcal{B}_k(w)$ also has low loss. A flat solution therefore identifies not a single hypothesis but an entire ball of them, and the log-volume of that ball is exactly what the complexity term pays for.

Theorem D.4 (Hamming-Ball Compression Bound)**.** *Suppose the Hamming-SAM solution $\hat{w}$ satisfies $\mathcal{L}_{\mathrm{adv}}(\hat{w}) \le \epsilon$ on a training set of size m, and let $Q_{\hat{w}}$ denote the uniform distribution on $\mathcal{B}_k(\hat{w})$. Then for any $\delta \in (0,1)$, with probability at least $1-\delta$:*

$$\mathbb{E}_{w' \sim Q_{\hat{w}}}\left[\mathcal{L}(w')\right] \;\le\; \epsilon + \sqrt{\frac{n\log 2 - \log V_k + \log(2\sqrt{m}/\delta)}{2m}}, \qquad V_k = \sum_{j=0}^{k} \binom{n}{j}. \tag{20}$$

Proof. Take the prior P_0 to be uniform on $\{-1,+1\}^n$, which is independent of the training set, and the posterior $Q_{\hat{w}}$ to be uniform on $\mathcal{B}_k(\hat{w})$. Both are supported on the hypercube, and

$$\mathrm{KL}(Q_{\hat{w}} \| P_0) \;=\; \sum_{w' \in \mathcal{B}_k(\hat{w})} \frac{1}{V_k} \log \frac{1/V_k}{2^{-n}} \;=\; \log \frac{2^n}{V_k} \;=\; n\log 2 - \log V_k.$$

The adversarial guarantee gives $\hat{\mathcal{L}}(w') \le \epsilon$ for *every* $w' \in \mathcal{B}_k(\hat{w})$, hence $\mathbb{E}_{w' \sim Q_{\hat{w}}}[\hat{\mathcal{L}}(w')] \le \epsilon$. Substituting both into the PAC-Bayes–McAllester inequality [40] yields (20). □

Three remarks follow.

The object bounded. The statement controls the Gibbs predictor drawn uniformly from the flat basin. A posterior spread over the ball is what makes the V_k saving available, since a point-mass posterior against a uniform prior has $\mathrm{KL} = n\log 2$. Drawing $w' \sim Q_{\hat{w}}$ is also the deployed model after up to k bit-flips, so (20) bounds the loss under bit-level corruption of the weights, a realistic failure mode for binary networks on edge hardware. If the population loss varies by at most ξ over $\mathcal{B}_k(\hat{w})$, the same bound holds for $\mathcal{L}(\hat{w})$ up to ξ, and Theorem J.2 bounds $\mathcal{L}(\hat{w})$ directly, with a point-mass posterior and a data-dependent prior.

No covering is needed. The reduction comes from the posterior being spread over V_k configurations that the adversarial guarantee certifies simultaneously, so the flatness of $\hat{w}$ is what pays for the complexity saving.

Roles of the three justifications. Theorem 4.1 and (20) both control a perturbed loss and motivate the objective, differing in the perturbation distribution (Bernoulli flips versus uniform on the ball). Theorem J.2 supplies the deterministic guarantee. The three are complementary in what they bound, not merely in how they are proved.

The term $\log V_k$ is the complexity saving gained from flatness, since the bound counts a ball of configurations rather than a single point. Because $\log V_k \approx k\log(n/k)$ grows more slowly than $n\log 2$, the saving is small in relative terms at the scales we study (for ResNet-18 with $k = 500$, roughly 5×10^3 nats against 7.7×10^6), and with a uniform prior the bound is vacuous when $n \gg m$. The data-dependent prior of § J counts complexity per adapted bit instead.

Corollary D.5 (Hamming-SAM Reduces Effective Hypothesis Complexity)**.** *Minimizing $\mathcal{L}_{\mathrm{adv}}(w)$ with Hamming radius k replaces the complexity $n\log 2$ of an arbitrary Boolean configuration by $n\log 2 - \log V_k$. For $k \ll n$, Stirling's approximation gives $\log V_k \approx k\log(n/k) + k - \frac{1}{2}\log(2\pi k)$ nats.*

D.3 An Equivalent Perspective: Distributionally Robust Optimization

The min-max objective (1) can be written as a Distributionally Robust Optimization (DRO) problem, which connects Hamming-SAM to the robust-optimization literature.

Proposition D.6 (Hamming-SAM as DRO)**.** *The Hamming-SAM objective is equivalent to a DRO problem over discrete ambiguity sets:*

$$\mathcal{L}_{\mathrm{adv}}(w) = \max_{w' \in \mathcal{B}_k(w)} \hat{\mathcal{L}}(w') = \max_{\mu \in \Delta(\mathcal{B}_k(w))} \mathbb{E}_{w' \sim \mu}\big[\hat{\mathcal{L}}(w')\big], \tag{21}$$

where $\Delta(\mathcal{B}_k(w))$ denotes the set of all probability distributions supported on $\mathcal{B}_k(w)$. Moreover, the maximizing distribution μ^* is the Dirac measure $\mu^* = \delta_{w^*}$ where $w^* = \arg\max_{w' \in \mathcal{B}_k(w)} \hat{\mathcal{L}}(w')$.

Proof. For any $\mu \in \Delta(\mathcal{B}_k(w))$, $\mathbb{E}_{w' \sim \mu}[\hat{\mathcal{L}}(w')] \leq \max_{w' \in \mathcal{B}_k(w)} \hat{\mathcal{L}}(w') = \mathcal{L}_{\mathrm{adv}}(w)$. Equality is achieved by $\mu^* = \delta_{w^*}$, which is a valid element of $\Delta(\mathcal{B}_k(w))$ since $w^* \in \mathcal{B}_k(w)$. □

This perspective gives a direct connection to the robustness literature. Hamming-SAM guards against the *worst-case realization* within the ambiguity set $\mathcal{B}_k(w)$. In the continuous DRO framework, this set corresponds to a Wasserstein ball, which is here replaced by its natural discrete counterpart, the Hamming ball. Given a uniform deviation bound between empirical and population risk on the ball, the adversarial loss upper-bounds the population risk of the minimizer.

Theorem D.7 (DRO Generalization via Hamming Ambiguity Sets)**.** *Let $\hat{w}$ minimize $\mathcal{L}_{\mathrm{adv}}(w)$ over $\{-1,+1\}^n$, and let $\mathcal{L}(w) = \mathbb{E}_{(x,y)\sim\mathcal{P}}[\ell(f_w(x), y)]$ denote the population risk. Suppose the sample loss satisfies $|\hat{\mathcal{L}}(w) - \mathcal{L}(w)| \leq \epsilon_m$ uniformly over $\mathcal{B}_k(\hat{w})$ with probability at least $1-\delta$. Then:*

$$\mathcal{L}(\hat{w}) \;\leq\; \mathcal{L}_{\mathrm{adv}}(\hat{w}) + \epsilon_m \;\leq\; \min_{w \in \{-1,+1\}^n} \mathcal{L}_{\mathrm{adv}}(w) + \epsilon_m. \tag{22}$$

In particular, if there exists a "true" parameter $w^\star$ with $\mathcal{L}(w') \leq \epsilon^\star$ for all $w' \in \mathcal{B}_k(w^\star)$, then $\mathcal{L}(\hat{w}) \leq \epsilon^\star + 2\epsilon_m$.

Proof. The first inequality is immediate: $\mathcal{L}(\hat{w}) = \hat{\mathcal{L}}(\hat{w}) + [\mathcal{L}(\hat{w}) - \hat{\mathcal{L}}(\hat{w})] \leq \hat{\mathcal{L}}(\hat{w}) + \epsilon_m \leq \mathcal{L}_{\mathrm{adv}}(\hat{w}) + \epsilon_m$, since $\hat{w} \in \mathcal{B}_k(\hat{w})$. The second follows because $\hat{w}$ minimizes $\mathcal{L}_{\mathrm{adv}}$.

For the "true parameter" claim, note $\mathcal{L}_{\mathrm{adv}}(w^\star) = \max_{w' \in \mathcal{B}_k(w^\star)} \hat{\mathcal{L}}(w') \leq \max_{w' \in \mathcal{B}_k(w^\star)} [\mathcal{L}(w') + \epsilon_m] \leq \epsilon^\star + \epsilon_m$. Since $\hat{w}$ minimizes $\mathcal{L}_{\mathrm{adv}}$, $\mathcal{L}_{\mathrm{adv}}(\hat{w}) \leq \mathcal{L}_{\mathrm{adv}}(w^\star) \leq \epsilon^\star + \epsilon_m$. Combining: $\mathcal{L}(\hat{w}) \leq \epsilon^\star + 2\epsilon_m$. □

E Choosing the Hamming Radius k: Full Analysis

The Hamming radius k is the central hyperparameter of Hamming-SAM, controlling both the strength of the adversarial probe and the tightness of the generalization guarantee. We now provide a formal analysis of its effect on the generalization bound, derive the optimal k in closed form for the compression bound, and give practical guidelines.

E.1 The Bias–Sharpness Decomposition

The generalization bounds of § 4 all share a common structure. The population loss is bounded by the sum of a *sharpness term* (which increases with k) and a *complexity term* (which decreases with k). To make this tradeoff explicit, we decompose the compression bound (20) as:

$$\mathbb{E}_{w' \sim Q_{\hat{w}}}\big[\mathcal{L}(w')\big] \;\leq\; \underbrace{\mathcal{L}_{\mathrm{adv}}(\hat{w}; k)}_{\text{sharpness: } \uparrow \text{ in } k} + \underbrace{\sqrt{\frac{n \log 2 - \log V_k + \log(2\sqrt{m}/\delta)}{2m}}}_{\text{complexity: } \downarrow \text{ in } k}, \tag{23}$$

where $V_k = \sum_{j=0}^{k} \binom{n}{j}$ is the Hamming ball volume, $Q_{\hat{w}}$ is uniform on $\mathcal{B}_k(\hat{w})$, and we write $\mathcal{L}_{\mathrm{adv}}(\hat{w}; k)$ to emphasize the dependence on k.

Small k (under-regularization). When $k = 0$, the Hamming ball is a singleton and $\mathcal{L}_{\mathrm{adv}}(w; 0) = \hat{\mathcal{L}}(w)$, so Hamming-SAM reduces to standard empirical risk minimization. The complexity term is $\sqrt{n \log 2/(2m)}$, which is vacuous for $n \gg m$. The optimizer can find sharp minima that happen to have low training loss but generalize poorly.

Large k (over-regularization). As k grows, $\mathcal{L}_{\mathrm{adv}}(w; k)$ increases because the adversary can search over more configurations. In the extreme $k = n$, the Hamming ball is the entire hypercube and $\mathcal{L}_{\mathrm{adv}}(w; n) = \max_{w'} \hat{\mathcal{L}}(w')$ for all w, so the objective becomes constant and uninformative. The complexity term vanishes ($\log V_n = n \log 2$), but the sharpness term is maximally pessimistic.

E.2 Optimal k from the Compression Bound

To find the value of k that minimizes the right-hand side of (23), we must model how $\mathcal{L}_{\mathrm{adv}}(w; k)$ grows with k. We consider a natural "smoothness" assumption on the discrete loss landscape.

***Definition* E.1** ((L, σ)-Smooth Boolean Landscape)**.** *The empirical risk $\hat{\mathcal{L}}$ is (L, σ)-smooth on the Boolean hypercube if, for all $w \in \{-1, +1\}^n$ and all $k \geq 1$, $\mathcal{L}_{\mathrm{adv}}(w; k) - \hat{\mathcal{L}}(w) \leq L \cdot k^{\sigma}$, where $L > 0$ and $\sigma > 0$.*

The exponent σ captures the local geometry. For $\sigma < 1$, the worst-case loss grows sublinearly, because most additional flips hit redundant directions. For $\sigma > 1$, it grows superlinearly because of positive interactions, as in the pairwise term $\binom{k}{2} h$ of Remark 5.9, which gives $\sigma = 2$.

Proposition E.2 (Approximate Optimal k for the Compression Bound)**.** *Under (L, σ)-smoothness with $k \ll n$, the leading-order approximation (24) of the right-hand side of (23) has the stationary point*

$$k^* \approx \begin{cases} \left(\dfrac{\log n}{2L\sigma\sqrt{2mn \log 2}} \right)^{\frac{1}{\sigma - 1}}, & \text{if } \sigma \neq 1, \\ n \exp\left(-2L\sqrt{2mn \log 2} \right), & \text{if } \sigma = 1. \end{cases}$$

For $\sigma \geq 1$ it is a local minimum. For $\sigma < 1$ it is a local maximum whenever $\log(n/k^*) > 1 + 1/(1 - \sigma)$, and the bound is then minimized at the boundary of the admissible range of k.

Proof. For $k \ll n$, Stirling's approximation (see, e.g., [18]) gives $\log V_k = \log \sum_{j=0}^{k} \binom{n}{j} \approx k \log(n/k)$ (dominated by the largest term $\binom{n}{k}$). Substituting the smoothness assumption into (23):

$$\mathrm{RHS}(k) \approx \hat{\mathcal{L}}(w) + Lk^{\sigma} + \sqrt{\frac{n \log 2 - k \log(n/k)}{2m}}. \tag{24}$$

Differentiating with respect to k and setting to zero:

$$\frac{d}{dk}\mathrm{RHS}(k) = L\sigma k^{\sigma - 1} - \frac{\log(n/k) - 1}{2\sqrt{2m(n \log 2 - k \log(n/k))}} = 0.$$

For the leading-order behavior (neglecting the -1 and the $k \log(n/k)$ in the denominator relative to $n \log 2$):

$$L\sigma k^{\sigma - 1} \approx \frac{\log(n/k)}{2\sqrt{2mn \log 2}}.$$

Solving for k with $\log(n/k) \approx \log n$ for $k \ll n$:

$$k^{\sigma - 1} = \frac{\log n}{2L\sigma\sqrt{2mn \log 2}}. \tag{25}$$

For $\sigma \neq 1$, raising both sides to the power $1/(\sigma - 1)$:

$$k^* \approx \left(\frac{\log n}{2L\sigma\sqrt{2mn \log 2}} \right)^{1/(\sigma - 1)}. \tag{26}$$

□

For $\sigma = 1$ the first-order condition reads $L = (\log(n/k) - 1)/(2\sqrt{2mn \log 2})$, whose solution is the stated value at leading order. For the second-order behavior, $\mathrm{RHS}''(k) \approx L\sigma(\sigma-1)k^{\sigma-2} + 1/(2k\sqrt{2mn \log 2})$. Substituting the first-order condition $L\sigma k^{\sigma-1} = (\log(n/k)-1)/(2\sqrt{2mn \log 2})$ gives

$$\mathrm{RHS}''(k^*) \approx \frac{(\sigma-1)\big(\log(n/k^*) - 1\big) + 1}{2k^*\sqrt{2mn \log 2}},$$

which is positive for $\sigma \geq 1$ (since $\log(n/k^*) > 1$ when $k^* \ll n$) and negative for $\sigma < 1$ under the stated condition.

Since L and σ are landscape-dependent quantities that are not known a priori, this formula serves as a qualitative description of the trade-off rather than a numerical prescription. The complexity term itself suggests a scale-free parametrization of the radius. For $k = \alpha n$ with $\alpha \leq 1/2$, the standard entropy bounds on binomial sums give $\log V_k = n\, H(\alpha) + O(\log n)$, where H is the binary entropy in nats, so the Hamming ball removes a fraction $H(\alpha)/\log 2$ of the description length of $\{-1, +1\}^n$ regardless of the network size. This motivates setting $k = \lfloor \alpha \cdot n \rfloor$ when applying the method across model sizes (§ 4.4).

E.3 Volume of the Hamming Ball

The combinatorial structure of the Hamming ball plays a central role in the analysis. The following bounds are useful for understanding the behavior of k.

Lemma E.3 (Hamming Ball Volume Bounds)**.** *For $1 \leq k \leq n/2$:*

$$\left(\frac{n}{k}\right)^k \;\leq\; V_k = \sum_{j=0}^{k} \binom{n}{j} \;\leq\; \left(\frac{en}{k}\right)^k. \tag{27}$$

In particular, $\log V_k = k \log(n/k) + \Theta(k)$.

Proof. For the upper bound, take any $x \in (0, 1]$ and note that $x^{j-k} \geq 1$ for every $j \leq k$, so

$$V_k \;=\; \sum_{j=0}^{k} \binom{n}{j} \;\leq\; \sum_{j=0}^{n} \binom{n}{j} x^{j-k} \;=\; x^{-k}(1+x)^n.$$

Choosing $x = k/n$ and using $(1 + k/n)^n \leq e^k$ gives $V_k \leq (n/k)^k e^k = (en/k)^k$. The lower bound follows from $V_k \geq \binom{n}{k} \geq (n/k)^k$. □

This lemma shows how the complexity saving grows with k. For $k = 1$ (single-bit perturbation), it is merely $\log n$ nats. For $k = \sqrt{n}$ it is $\Theta(\sqrt{n} \log \sqrt{n})$, larger but still a vanishing fraction of $n \log 2$. Only for $k = \Theta(n)$ does the saving become comparable to the total. At $k = n/2$, $\log V_k \approx n \log 2 - O(1)$, nearly eliminating the complexity term, though $\mathcal{L}_{\mathrm{adv}}$ is then uninformative. The "sweet spot" lies in the range $k = \Theta(\sqrt{n})$ to $k = \Theta(n^{2/3})$, where the complexity reduction is large enough to matter but the Hamming ball is still small enough that $\mathcal{L}_{\mathrm{adv}}$ remains informative.

E.4 Relationship to Continuous SAM's ρ

The Hamming radius k is the Boolean analogue of the perturbation radius ρ in continuous SAM [19]. The following proposition makes this correspondence quantitative.

Proposition E.4 (Correspondence Between k and ρ)**.** *Consider a continuous weight vector $\tilde{w} \in \mathbb{R}^n$ and its binarization $w = \operatorname{sign}(\tilde{w})$. A perturbation ϵ with $\|\epsilon\|_2 = \rho$ flips, on average,*

$$\mathbb{E}\big[d_H(\operatorname{sign}(\tilde{w}), \operatorname{sign}(\tilde{w}+\epsilon))\big] \;=\; \sum_{i=1}^{n} \Phi\left(\frac{-|\tilde{w}_i|}{\rho/\sqrt{n}}\right), \tag{28}$$

where Φ is the standard normal CDF and ϵ is drawn uniformly on the ℓ_2-sphere of radius ρ. For the special case of "marginally confident" weights $|\tilde{w}_i| = c$ for all i (constant magnitude), this simplifies to:

$$k_{\text{equiv}} \;=\; n \cdot \Phi\left(\frac{-c\sqrt{n}}{\rho}\right) \;\approx\; n \cdot \frac{\rho}{c\sqrt{2\pi n}} \exp\left(-\frac{c^2 n}{2\rho^2}\right) \tag{29}$$

for $c\sqrt{n} \gg \rho$ (the "confident weight" regime where continuous SAM suffers from the vanishing perturbation effect).

Proof. Each coordinate ϵ_i of a uniform point on the ℓ_2-sphere has marginal distribution $\mathcal{N}(0, \rho^2/n)$. Bit i flips when $\operatorname{sign}(\tilde{w}_i + \epsilon_i) \neq \operatorname{sign}(\tilde{w}_i)$, which requires $|\epsilon_i| > |\tilde{w}_i|$ with opposite sign. This has probability $\Phi(-|\tilde{w}_i|/(\rho/\sqrt{n}))$. Summing over i and using linearity of expectation gives (28). The Gaussian tail approximation $\Phi(-x) \approx \phi(x)/x$ for $x \gg 1$ yields (29). □

Remark E.5 (The vanishing perturbation effect, quantified). Equation (29) shows the exponential decay of k_{equiv} in the ratio $c\sqrt{n}/\rho$. For a typical trained network with $c = 2$ (moderate confidence), $n = 10^6$, and the standard SAM choice $\rho = 0.05$, we obtain $c\sqrt{n}/\rho = 2000/0.05 = 40{,}000$, giving $k_{\text{equiv}} \approx 0$, so continuous SAM flips essentially zero bits. In contrast, Hamming-SAM with $k = 100$ always probes a 100-dimensional subspace. This makes precise the vanishing perturbation argument of § 3.

E.5 Practical Guidelines

Based on the theoretical analysis above and preliminary experiments, we recommend the following approach for setting k:

1. **Starting point:** A radius between $10^{-5}n$ and $10^{-3}n$ is a reasonable starting range for tuning. Our experiments use a fixed grid of absolute radii $\{50, 100, 500, 1000\}$.
2. **Validation-based tuning:** Perform a grid search, e.g. $k \in \{n/100000,\ n/10000,\ n/1000\}$ (i.e., $\alpha \in \{10^{-5}, 10^{-4}, 10^{-3}\}$), monitoring the generalization gap on a held-out set. The optimal k typically lies in the range where the sharpness $B_k = \mathcal{L}_{\text{adv}}(w; k) - \hat{\mathcal{L}}(w)$ is non-trivial (say, $B_k \in [0.01, 0.5]$ for cross-entropy loss) but not so large that the adversarial loss dominates. In our experiments we instead search a fixed grid of absolute radii, $k \in \{50, 100, 500, 1000\}$, which spans α from roughly 3×10^{-6} to 2×10^{-3} across the models considered.
3. **Sharpness-calibrated k:** Monitor B_k during training. If $B_k < 0.01$ persistently, k is too small and the optimizer is not probing enough of the landscape. If $B_k > 1.0$ persistently, k is too large and the adversarial probe is finding configurations far outside the relevant basin. Adjust k to maintain B_k in a moderate range, or use the adaptive schedule $k(t) = \max(1, \lfloor k_0 e^{-\lambda t} \rfloor)$.

F Details on the Connection to the Discrete Interaction Hessian

A second-order analysis gives more insight into the curvature of the discrete loss landscape. In continuous optimization, sharpness is characterized by the eigenvalues of the Hessian $\nabla^2 \mathcal{L}$ [39]. For Boolean networks, the analogous object is the *discrete interaction Hessian*, whose entries capture the pairwise interaction between bit-flips.

***Definition* F.1** (Discrete Interaction Hessian)**.** *For an empirical risk $\hat{\mathcal{L}}$ and Boolean weight vector $w \in \{-1, +1\}^n$, the entry $H_{ij}(w)$ is defined as:*

$$H_{ij}(w) = \hat{\mathcal{L}}(w^{(ij)}) - \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w^{(j)}) + \hat{\mathcal{L}}(w), \tag{30}$$

where $w^{(i)}$ denotes w with the i-th bit flipped, and $w^{(ij)}$ denotes w with both bits i and j flipped.

This quantity is the second-order flip difference of a pseudo-Boolean function, a classical object in combinatorial optimization [10, 5]. In terms of the partial derivative, $H_{ij}(w) = w_i w_j \big[f_i'(w|_{w_j=+1}) - f_i'(w|_{w_j=-1})\big]$, that is, $w_i w_j$ times the second-order partial derivative. At w^t it is the pairwise Möbius coefficient of the set function f (Lemma G.5). Its average over all contexts $S \subseteq [n] \setminus \{i,j\}$ is the Banzhaf interaction index of the pair [21], which equals $4\, w_i^t w_j^t\, \widehat{\mathcal{L}}(\{i,j\})$, with $\widehat{\mathcal{L}}(\{i,j\})$ the Walsh–Hadamard coefficient in the normalization of [43]. The interaction Hessian is the pointwise quantity, and it coincides with these global coefficients only when $\hat{\mathcal{L}}$ has no interactions of order three or more involving i and j.

The following lemma clarifies the relationship between the discrete Hessian of $\hat{\mathcal{L}}$ and the second-order structure of the risk-increase set function f of § 5.1.

Lemma F.2 (Hessian of $\hat{\mathcal{L}}$ as Second-Order Derivative of f)**.** *Let $f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$ be the risk-increase set function. For any $S \subseteq [n]$ and $i, j \notin S$ with $i \neq j$, the second-order discrete derivative of f at context S equals the interaction Hessian of $\hat{\mathcal{L}}$ evaluated at $u = w^t \oplus \mathbf{1}_S$:*

$$\Delta_{ij} f(S) \;\triangleq\; f(S \cup \{i,j\}) - f(S \cup \{i\}) - f(S \cup \{j\}) + f(S) \;=\; H_{ij}(w^t \oplus \mathbf{1}_S). \tag{31}$$

Proof. Substituting the definition $f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$ into each of the four terms:

$$\begin{aligned}\Delta_{ij} f(S) = &\big[\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_{S\cup\{i,j\}}) - \hat{\mathcal{L}}(w^t)\big] - \big[\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_{S\cup\{i\}}) - \hat{\mathcal{L}}(w^t)\big] \\ &- \big[\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_{S\cup\{j\}}) - \hat{\mathcal{L}}(w^t)\big] + \big[\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)\big].\end{aligned} \tag{32}$$

The four copies of $-\hat{\mathcal{L}}(w^t)$ cancel. Writing $u = w^t \oplus \mathbf{1}_S$ and noting that $w^t \oplus \mathbf{1}_{S\cup\{i\}} = u^{(i)}$, $w^t \oplus \mathbf{1}_{S\cup\{j\}} = u^{(j)}$, and $w^t \oplus \mathbf{1}_{S\cup\{i,j\}} = u^{(ij)}$:

$$\Delta_{ij} f(S) = \hat{\mathcal{L}}(u^{(ij)}) - \hat{\mathcal{L}}(u^{(i)}) - \hat{\mathcal{L}}(u^{(j)}) + \hat{\mathcal{L}}(u) = H_{ij}(u). \qquad \square$$

This identity shows that H_{ij} of the empirical loss, evaluated at the "context point" $w^t \oplus \mathbf{1}_S$, is exactly the pairwise interaction of f at context S. The constant $\hat{\mathcal{L}}(w^t)$ in the definition of f drops out because second-order differences annihilate additive constants. The same structure reappears in § G, where the Möbius expansion of f separates its first- and second-order terms and thereby bounds the deficit of the one-shot top-k ascent. The connection to submodularity follows directly.

Proposition F.3 (Hessian–Submodularity Connection)**.** *The set function $f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t)$ is submodular if and only if $H_{ij}(w^t \oplus \mathbf{1}_S) \leq 0$ for all $S \subseteq [n]$ and all $i, j \notin S$ with $i \neq j$. Equivalently, negative off-diagonal entries correspond to diminishing returns (flipping both bits gives less loss increase than the sum of individual flips), and positive entries correspond to reinforcing interactions (flipping both gives more).*

Proof. The submodularity condition requires $f(S \cup \{i\}) - f(S) \geq f(S \cup \{i,j\}) - f(S \cup \{j\})$ for all S, i, j. Rearranging: $f(S \cup \{i,j\}) - f(S \cup \{i\}) - f(S \cup \{j\}) + f(S) \leq 0$. By Lemma F.2, the left-hand side equals $H_{ij}(w^t \oplus \mathbf{1}_S)$. $\square$

If $H_{ij}(w) = 0$ for all $i \neq j$, the loss function is modular and weights contribute independently. Positive values indicate reinforcing flips, and negative values indicate redundancy. Hamming-SAM convergence regions should therefore tend to have lower $|H_{ij}|$ than typical points, a prediction supported by the convergence condition $\eta > kh$ (Theorem 5.8), which favors regions with small h, and by the connection to γ_k in the ascent guarantee. § 7.2 and § N confirm this: on every architecture measured, the sampled $|H_{ij}|$ statistics at Hamming-SAM solutions lie below those at BOLD solutions at convergence.

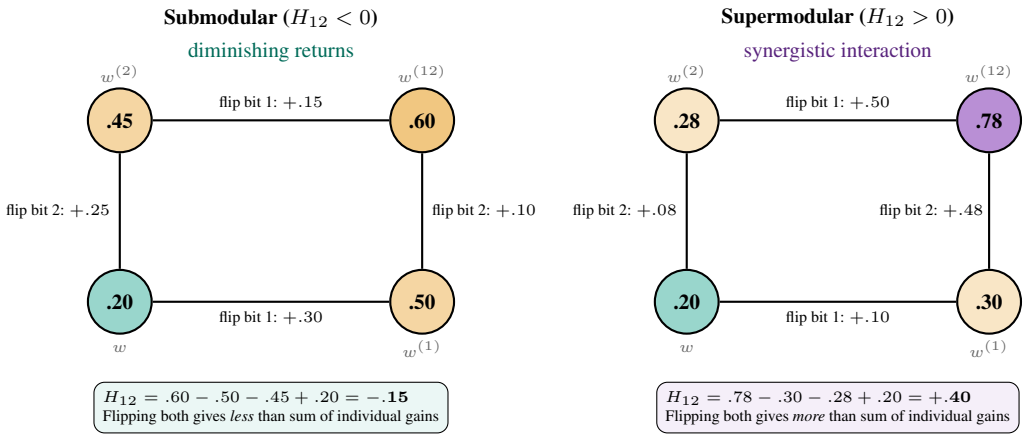


Figure 7: **Discrete interaction Hessian on $\{-1,+1\}^2$.** Each vertex shows a loss value. *Left:* Submodular case (H_{12}=-0.15): the marginal gain of flipping bit 2 decreases from 0.25 to 0.10 after bit 1 is flipped (diminishing returns). *Right:* Supermodular case (H_{12}=$+0.40$): the marginal gain of flipping bit 2 *increases* from 0.08 to 0.48 after bit 1 is flipped. In this case, the greedy ascent can underestimate the worst case.

Spectral Gap of the Hamming Graph. The flatness of a discrete minimum can further be quantified by the spectral gap of the graph Laplacian restricted to the loss-weighted Hamming neighborhood. The hypercube $\{-1,+1\}^n$ is a regular graph where nodes are configurations and edges connect those with Hamming distance one. It is well known that the eigenvalues of the hypercube Laplacian are $\{2j : j = 0, 1, \ldots, n\}$ with multiplicities $\binom{n}{j}$ [9]. A large spectral gap in the local subgraph weighted by the loss indicates a well-connected region where the loss changes slowly, which gives faster mixing and better robustness. By minimizing the adversarial loss in $\mathcal{B}_k(w)$, Hamming-SAM implicitly regularizes for these spectral properties.

G Detailed Proof of Approximation Guarantees for the Ascent Step

A key requirement for the Hamming-SAM framework is a guarantee that the greedy ascent identifies a near-worst-case configuration within $\mathcal{B}_k(w)$. In continuous SAM, the ascent is justified as a first-order Taylor approximation [19]. In the discrete domain, the quality of the greedy search depends on the approximate submodularity of the risk-increase function.

The problem of finding $\hat{w}$ is a cardinality-constrained maximization of the non-negative set function

$$f(S) = \hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S; \mathcal{D}) - \hat{\mathcal{L}}(w^t; \mathcal{D}), \quad S \subseteq [n],\ |S| \le k, \tag{33}$$

where $w^t \oplus \mathbf{1}_S$ denotes the configuration obtained by flipping all bits indexed by S. Note $f(\emptyset) = 0$ by definition. If f were monotone and submodular, the classical greedy algorithm would achieve an approximation ratio of $(1 - 1/e)$ [41]. However, neural networks exhibit complex interactions, where flipping two bits together may increase the loss far more than the sum of their individual effects. This violates submodularity.

We therefore use the framework of [3], which extends greedy guarantees to non-submodular functions using two parameters:

***Definition* G.1** (Submodularity Ratio [12])**.** *The submodularity ratio γ_k of a non-negative set function f with respect to a parameter k is the largest scalar γ such that, for all disjoint sets $L, S \subseteq [n]$ with $|L| \le k$:*

$$\sum_{j \in L} \big[f(S \cup \{j\}) - f(S)\big] \ \ge\ \gamma \cdot \big[f(S \cup L) - f(S)\big]. \tag{34}$$

***Definition* G.2** (Generalized Curvature [3])**.** *Let $\rho_j(A) = f(A \cup \{j\}) - f(A)$. The generalized curvature of a non-negative set function f with $f(\emptyset) = 0$ is the smallest $\alpha \in [0, 1]$ such that, for all $S, \Omega \subseteq \mathcal{V}_+$ and $i \in S \setminus \Omega$,*

$$\rho_i\big((S \setminus \{i\}) \cup \Omega\big) \ \ge\ (1 - \alpha)\, \rho_i(S \setminus \{i\}). \tag{35}$$

Equivalently, α bounds how much the marginal gain of an element can shrink when other elements are added to its context. For modular functions, $\alpha = 0$ (gains are constant). For submodular functions, α can be as large as 1 (gains can vanish).

The submodularity ratio $\gamma \in (0, 1]$ quantifies how close f is to submodular ($\gamma = 1$ for submodular functions). The curvature $\alpha \in [0, 1]$ measures the maximum rate at which marginal gains decrease ($\alpha = 0$ for modular functions, $\alpha = 1$ in the worst case).

Theorem 5.2 (Sequential Greedy Ascent Guarantee)**.** *Let f be the set function defined in (33), restricted to subsets of $\mathcal{V}_+$, with $f(\emptyset) = 0$, and suppose f is non-negative and non-decreasing (monotone) on these subsets. Let $\hat{S}$ be the set of k indices selected by the sequential greedy ascent, and $S^* = \arg\max_{S \subseteq \mathcal{V}_+, |S| \leq k} f(S)$ the optimal set. Then:*

$$f(\hat{S}) \;\geq\; \frac{1}{\alpha}\left[1 - e^{-\gamma_k \alpha}\right] f(S^*). \tag{36}$$

In terms of the adversarial loss:

$$\hat{\mathcal{L}}(\hat{w}) - \hat{\mathcal{L}}(w^t) \;\geq\; \frac{1}{\alpha}\left[1 - e^{-\gamma_k \alpha}\right]\left[\hat{\mathcal{L}}(w^*) - \hat{\mathcal{L}}(w^t)\right], \tag{37}$$

where $\hat{w} = w^t \oplus \mathbf{1}_{\hat{S}}$ and $w^* = w^t \oplus \mathbf{1}_{S^*}$.

Proof. Let $f : 2^{[n]} \to \mathbb{R}_{\geq 0}$ be the set function defined in (33) with $f(\emptyset) = 0$. The greedy algorithm constructs a sequence $S_0 = \emptyset \subset S_1 \subset \cdots \subset S_k$ where at step j, we add $i_j = \arg\max_{i \notin S_{j-1}}[f(S_{j-1} \cup \{i\}) - f(S_{j-1})]$.

Let $S^* = \arg\max_{|S| \leq k} f(S)$. By the definition of the submodularity ratio γ_k (Definition 5.1) applied to $L = S^* \setminus S_{j-1}$ and $S = S_{j-1}$ (noting $|L| \leq k$):

$$\sum_{i \in S^* \setminus S_{j-1}} \left[f(S_{j-1} \cup \{i\}) - f(S_{j-1})\right] \;\geq\; \gamma_k \left[f(S_{j-1} \cup S^*) - f(S_{j-1})\right]. \tag{38}$$

Since f is monotone, $f(S_{j-1} \cup S^*) \geq f(S^*)$, so:

$$f(S_{j-1} \cup S^*) - f(S_{j-1}) \;\geq\; f(S^*) - f(S_{j-1}). \tag{39}$$

The greedy choice i_j achieves at least $1/k$ of the sum in (38) (since $|S^* \setminus S_{j-1}| \leq k$). Combining (38) and (39):

$$f(S_j) - f(S_{j-1}) \;\geq\; \frac{\gamma_k}{k}\left[f(S^*) - f(S_{j-1})\right]. \tag{40}$$

Define $\delta_j = f(S^*) - f(S_j)$. Then:

$$\delta_j \;\leq\; \delta_{j-1} - \frac{\gamma_k}{k}\,\delta_{j-1} \;=\; \left(1 - \frac{\gamma_k}{k}\right)\delta_{j-1}.$$

Iterating k times from $\delta_0 = f(S^*)$:

$$\delta_k \;\leq\; \left(1 - \frac{\gamma_k}{k}\right)^k f(S^*) \;\leq\; e^{-\gamma_k}\, f(S^*).$$

Therefore:

$$f(\hat{S}) = f(S_k) \;\geq\; \left[1 - e^{-\gamma_k}\right] f(S^*). \tag{41}$$

Tighter bound with curvature α. The bound (41) uses only the submodularity ratio and monotonicity. A tighter result is obtained by additionally incorporating the generalized curvature α (Definition G.2), which bounds how much marginal gains decrease relative to singleton gains: $f(S \cup \{j\}) - f(S) \geq (1 - \alpha)\, f(\{j\})$ for all S and $j \notin S$. [3] show that this yields the refined bound:

$$f(\hat{S}) \;\geq\; \frac{1}{\alpha}\left(1 - e^{-\gamma_k \alpha}\right) f(S^*), \tag{42}$$

which is strictly tighter than (41) for $\alpha < 1$. For $\gamma_k = 1$ and $\alpha = 1$ (submodular), this recovers the classical $(1 - 1/e)$ bound of [41]. For $\alpha \to 0$ (modular), L'Hôpital's rule gives $f(\hat{S}) \geq \gamma_k\, f(S^*)$. We refer to [3] for the complete derivation. □

Remark G.4. When $\gamma_k = 1$ and $\alpha \to 0$ (the modular case), the bound recovers the exact optimum $f(\hat{S}) = f(S^*)$. When $\gamma_k = 1$ and $\alpha = 1$ (the submodular case), it recovers the classical $(1 - 1/e)$ guarantee of [41].

Assumptions and scope. Theorem 5.2 requires f to be non-negative and non-decreasing (monotone) on subsets of $\mathcal{V}_+$. Every bit in $\mathcal{V}_+$ increases the loss when flipped alone, but flipping other bits first changes its gain by one interaction entry per flip, so $\rho_i(S) = \mathrm{BV}_i(w^t \oplus \mathbf{1}_S) \geq \mathrm{BV}_i(w^t) - |S|\,h$, with h the interaction bound of Theorem 5.4. Hence f is monotone on subsets of $\mathcal{V}_+$ of size at most K whenever $\min_{i\in\mathcal{V}_+} \mathrm{BV}_i(w^t) > (K-1)h$, which is the precise form of Remark 5.3. Monotonicity enters the proof through the inequality $f(S_{j-1} \cup S^*) \geq f(S^*) + (1-\alpha)\, f(S_{j-1})$, obtained by adding the elements of the greedy sequence to S^* one at a time. When f is not monotone, this step may fail and the ratio to $f(S^*)$ is no longer guaranteed. The greedy's own output $f(\hat{S})$ is unaffected. Empirical measurement of γ_k during training (§ 7.2) quantifies how close practical landscapes are to the submodular regime.

G.1 Guarantee for the One-Shot Top-k Ascent

This subsection develops the approximation guarantee for the one-shot top-k ascent that Algorithm 1 actually implements. The result is not a strengthening of Theorem 5.2 but a complement to it. Theorem 5.2 bounds the sequential greedy under approximate submodularity, and is therefore governed by the *sign* structure of the interaction Hessian, since $\gamma_k = 1$ if and only if $H_{ij} \leq 0$ (Proposition F.3). The result below bounds the top-k rule under a bound on interaction *magnitude*, $|H_{ij}| \leq h$, and requires neither monotonicity nor submodularity. Neither statement implies the other. § 5.1 sets out their respective roles, and § N.4 measures the empirical gap between the two procedures.

Throughout this subsection we color the first-order (Boolean Variation), pairwise (interaction Hessian) and higher-order residual contributions consistently, so that the role of each can be tracked across the expansion, the bound and its corollaries.

The two procedures also differ in cost. The sequential greedy re-evaluates marginal gains after each selection and therefore requires k backward passes per optimization step, which is intractable at the radii we use. The top-k rule obtains all n marginal gains in a single backward pass. The theorem below quantifies the cost of this speed-up.

Higher-order discrete derivatives. For $T \subseteq [n]$ with $|T| \geq 1$, let $\Delta_T(w^t)$ denote the order-$|T|$ discrete derivative of $\hat{\mathcal{L}}$ at w^t, defined by Möbius inversion of f:

$$\Delta_T(w^t) \;=\; \sum_{U\subseteq T} (-1)^{|T|-|U|} f(U). \tag{43}$$

The first two orders recover the quantities already in use, $\Delta_{\{i\}}(w^t) = \mathrm{BV}_i(w^t)$ and $\Delta_{\{i,j\}}(w^t) = H_{ij}(w^t)$, by Definition F.1. We write $h_p = \max_{w,\,|T|=p} |\Delta_T(w)|$ for the largest order-p interaction, so that $h_2 = h$.

Lemma G.5 (Möbius Expansion of the Risk-Increase Function)**.** *For every $S \subseteq [n]$,*

$$f(S) \;=\; \sum_{i\in S} \mathrm{BV}_i(w^t) \;+\; \sum_{\{i,j\}\subseteq S} H_{ij}(w^t) \;+\; R(S), \qquad |R(S)| \;\leq\; \sum_{p=3}^{|S|} \binom{|S|}{p} h_p. \tag{44}$$

Proof. Any set function with $f(\emptyset) = 0$ admits the unique multilinear (Möbius) representation $f(S) = \sum_{\emptyset\neq T\subseteq S} \Delta_T(w^t)$ with coefficients given by (43) [10]. Partitioning the sum by $|T| \in \{1\}$, $\{2\}$ and $\{3,\ldots,|S|\}$ gives the three terms of (44), where the order-one coefficients are the Boolean Variations and the order-two coefficients are the interaction Hessian entries. The residual $R(S)$ collects the remaining $\sum_{p\geq 3} \binom{|S|}{p}$ coefficients, each bounded in magnitude by h_p, and the triangle inequality gives the stated bound. □

Theorem 5.4 (Top-k Ascent Guarantee)**.** *Let $\hat{S}_{\mathrm{tk}}$ be the set selected by the top-k rule of Algorithm 1, restricted to bits with $\mathrm{BV}_i > 0$, and let $S^* = \arg\max_{|S|\leq k} f(S)$. Suppose $|H_{ij}(w)| \leq h$ for all w and $i \neq j$, and $|R(S)| \leq R_{\max}$ for all $|S| \leq k$. Then*

$$f(\hat{S}_{\mathrm{tk}}) \;\geq\; f(S^*) \;-\; k(k-1)\,h \;-\; 2R_{\max}. \tag{45}$$

Proof. Apply Lemma G.5 to S^* and to $\hat{S}_{\mathrm{tk}}$ and subtract:

$$f(S^*)-f(\hat{S}_{\mathrm{tk}}) = \underbrace{\Big[\sum_{i\in S^*}\mathrm{BV}_i - \sum_{i\in \hat{S}_{\mathrm{tk}}}\mathrm{BV}_i\Big]}_{(\mathrm{A})\le 0} + \underbrace{\Big[\sum_{\{i,j\}\subseteq S^*} H_{ij} - \sum_{\{i,j\}\subseteq \hat{S}_{\mathrm{tk}}} H_{ij}\Big]}_{(\mathrm{B})\le k(k-1)h} + \underbrace{\big[R(S^*) - R(\hat{S}_{\mathrm{tk}})\big]}_{(\mathrm{C})\le 2R_{\max}}. \tag{46}$$

Term (A), first order. Let $P = \{i \in S^* : \mathrm{BV}_i > 0\}$. Discarding the non-positive entries of S^* can only increase the sum, so $\sum_{i\in S^*}\mathrm{BV}_i \le \sum_{i\in P}\mathrm{BV}_i$. Since $|P| \le |S^*| \le k$ and $\hat{S}_{\mathrm{tk}}$ consists of the k largest strictly positive Boolean Variations, $\sum_{i\in P}\mathrm{BV}_i \le \sum_{i\in \hat{S}_{\mathrm{tk}}}\mathrm{BV}_i$. Hence $(\mathrm{A}) \le 0$. The restriction to $\mathrm{BV}_i > 0$ is needed here: without it, a top-k set padded with negative entries could have a smaller sum than a strictly smaller optimal set.

Term (B), pairwise. Each expansion contains at most $\binom{k}{2}$ pairwise coefficients, each of magnitude at most h, so $(\mathrm{B}) \le 2\binom{k}{2}h = k(k-1)h$.

Term (C), higher order. By assumption $|R(S)| \le R_{\max}$ for both sets, so $(\mathrm{C}) \le 2R_{\max}$ by the triangle inequality.

Summing the three bounds gives $f(S^*) - f(\hat{S}_{\mathrm{tk}}) \le k(k-1)h + 2R_{\max}$, which is (45). □

Remark G.7 (Relation to Theorem 5.2). The bound is *additive* in the interaction strength, whereas Theorem 5.2 is *multiplicative* in the submodularity ratio. The two are complementary. Theorem 5.2 certifies a constant fraction of the optimum however large $f(S^*)$ is, but needs monotonicity and degrades as $\gamma_k \to 0$. Theorem 5.4 needs neither, but is informative only while $k(k-1)h$ stays small relative to $f(S^*)$. The ascent is therefore near-optimal whenever *either* the landscape is approximately submodular *or* its pairwise interactions are weak.

Remark G.8 (Size of the residual). The residual collects interactions of order three and higher and enters Theorem 5.4 only through the assumed bound $R_{\max}$. If higher-order interactions decay as $h_p \le h^{p-1}$ for $p \ge 3$, the leading residual term is $\binom{k}{3}h^2 = O(k^3h^2)$, which is dominated by the pairwise term $k(k-1)h$ whenever $kh \ll 1$.

Inexact oracles. § 4.2 notes that the backward pass returns a first-order estimate $\widetilde{\mathrm{BV}}_i$ rather than the exact finite difference. The guarantee degrades gracefully in the oracle error.

Proposition G.9 (Top-k Ascent with an Inexact Oracle)**.** *Suppose the ascent ranks bits by a signal $\widetilde{\mathrm{BV}}$ with $\max_i |\widetilde{\mathrm{BV}}_i - \mathrm{BV}_i| \le \varepsilon$, and let $\hat{S}_{\mathrm{tk}}$ be the resulting selection, restricted to bits with $\widetilde{\mathrm{BV}}_i > 0$. Then under the assumptions of Theorem 5.4,*

$$f(\hat{S}_{\mathrm{tk}}) \;\ge\; f(S^*) \;-\; 2k\varepsilon \;-\; k(k-1)h \;-\; 2R_{\max}. \tag{47}$$

Proof. Only term (A) changes. Let $P = \{i \in S^* : \mathrm{BV}_i > 0\}$ and $P^+ = \{i \in P : \widetilde{\mathrm{BV}}_i > 0\}$, so $|P^+| \le |P| \le k$. Then $\sum_{S^*}\mathrm{BV}_i \le \sum_P \mathrm{BV}_i \le \sum_P \widetilde{\mathrm{BV}}_i + k\varepsilon \le \sum_{P^+}\widetilde{\mathrm{BV}}_i + k\varepsilon \le \sum_{\hat{S}_{\mathrm{tk}}}\widetilde{\mathrm{BV}}_i + k\varepsilon \le \sum_{\hat{S}_{\mathrm{tk}}}\mathrm{BV}_i + 2k\varepsilon$, where the first step discards non-positive entries, the second and last use the oracle bound on at most k terms, the third discards entries with $\widetilde{\mathrm{BV}}_i \le 0$, and the fourth uses that $\hat{S}_{\mathrm{tk}}$ maximizes $\sum\widetilde{\mathrm{BV}}$ over positive-signal sets of size at most k. Hence $(\mathrm{A}) \le 2k\varepsilon$, and (B) and (C) are bounded as before. □

Controlling the adversarial objective. The same expansion bounds the sharpness itself, which is the quantity the outer minimization targets.

Remark G.10 (Sharpness Composition). Under the assumptions of Theorem 5.4, for every w,

$$\mathcal{L}_{\mathrm{adv}}(w) - \hat{\mathcal{L}}(w) \;=\; \max_{|S|\le k} f(S) \;\le\; k\,B_1(w) + \binom{k}{2}h + R_{\max}, \tag{48}$$

where $B_1(w) = \max_i \max(\mathrm{BV}_i(w), 0)$ is the one-bit sharpness.

Proof. Apply Lemma G.5 to an arbitrary S with $|S| \le k$. The first-order term satisfies $\sum_{i\in S} \mathrm{BV}_i \le \sum_{i\in S} \max(\mathrm{BV}_i, 0) \le kB_1(w)$, the second-order term is at most $\binom{k}{2}h$, and the residual is at most $R_{\max}$. Taking the maximum over S gives (48). □

We note two consequences. First, since Theorem 5.8 drives $\hat{\mathcal{L}}(w^t)$ down monotonically, (48) keeps $\mathcal{L}_{\mathrm{adv}}(w^t)$ within an explicit band above it, and the band is observable at run time because B_1 costs one backward pass. Full min-max convergence therefore reduces to controlling B_1, which Phase 2 targets directly. Second, at $k = 1$ both the pairwise and residual terms vanish identically and the bound reads $\mathcal{L}_{\mathrm{adv}}(w) - \hat{\mathcal{L}}(w) \le B_1(w)$, which is an identity, since the one-bit sharpness is by definition the largest loss increase over $\mathcal{B}_1(w)$. The same degeneracy explains why the deficit $k(k-1)h$ of Theorem 5.4 vanishes at $k = 1$. The top-1 rule is then exactly optimal, and the sequential and one-shot procedures coincide.

H Details of Convergence Analysis

We analyze convergence in two stages. We first study a simplified base-point descent variant with a clean Lyapunov guarantee, and then the full SAM-style algorithm with Phase 2 evaluated at the adversarial point.

***Definition* H.1** (η-Discrete Stationarity)**.** *A configuration $w \in \{-1, +1\}^n$ is η-stationary if for every $i \in [n]$:*

$$\mathrm{BV}_i(w) \triangleq \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w) \ge -\eta, \tag{49}$$

so that no single bit-flip can reduce the empirical loss by more than η. A configuration is exactly stationary *when $\eta = 0$.*

Note that this characterizes local minima of the empirical loss $\hat{\mathcal{L}}$, not of the adversarial loss $\mathcal{L}_{\mathrm{adv}}$. The overall Hamming-SAM objective (1) targets $\mathcal{L}_{\mathrm{adv}}$, and stationarity of $\hat{\mathcal{L}}$ is a necessary but not sufficient condition, since a point with low $\hat{\mathcal{L}}$ may still have high worst-case loss in $\mathcal{B}_k$. Theorems 5.7 and 5.8 guarantee convergence of $\hat{\mathcal{L}}(w^t)$, and Remark 5.9 bounds the remaining gap to $\mathcal{L}_{\mathrm{adv}}$ explicitly along the trajectory.

Theorem 5.7 (Finite-Time Convergence (Base-Point Descent))**.** *Suppose the empirical risk $\hat{\mathcal{L}} : \{-1, +1\}^n \to [0, B]$ is bounded. Consider a variant of Algorithm 1 where Phase 2 evaluates BV_i at w^t (rather than at the adversarial point $\hat{w}$), selecting $i^* = \arg\min_i \mathrm{BV}_i(w^t)$ and flipping only when $\mathrm{BV}_{i^*}(w^t) < -\eta$ for a threshold $\eta > 0$. Then this variant converges to an η-stationary point in at most*

$$T \le \left\lfloor \frac{B}{\eta} \right\rfloor \tag{50}$$

iterations.

Proof. Define the Lyapunov function $\Phi(w^t) = \hat{\mathcal{L}}(w^t)$.

Monotone decrease. At each iteration, Phase 2 accepts a bit-flip at position i^* only if $\mathrm{BV}_{i^*}(w^t) < -\eta$. Since $\hat{\mathcal{L}}(w^{t+1}) = \hat{\mathcal{L}}((w^t)^{(i^*)}) = \hat{\mathcal{L}}(w^t) + \mathrm{BV}_{i^*}(w^t) \le \hat{\mathcal{L}}(w^t) - \eta$, the Lyapunov function decreases by at least η at each accepted step.

Finite termination. Since $\hat{\mathcal{L}} \ge 0$ and $\Phi(w^0) \le B$, the sequence $\{\Phi(w^t)\}$ can decrease at most $\lfloor B/\eta \rfloor$ times, after which no further bit-flip satisfies the threshold condition, and the algorithm terminates at an η-stationary point. □

The base-point variant does not exploit Phase 1, since the descent direction is computed at w^t rather than at the adversarial point. We now show that the full SAM-style algorithm also converges, provided the discrete Hessian is bounded. The key observation is that the discrete Hessian measures how much the Boolean Variation signal changes when moving between $\hat{w}$ and w^t.

Theorem 5.8 (Finite-Time Convergence (SAM-Style Descent)). *Suppose $\hat{\mathcal{L}} : \{-1,+1\}^n \to [0, B]$ and the discrete interaction Hessian is bounded: $|H_{ij}(w)| \le h$ for all $w \in \{-1,+1\}^n$ and $i \neq j$. In Algorithm 1, suppose Phase 2 selects $i^* \notin \mathcal{I}$ (the descent bit is distinct from the ascent set) and accepts the flip only when $\mathrm{BV}_{i^*}(\hat{w}) < -\eta$ with $\eta > k \cdot h$. Then each accepted step satisfies*

$$\hat{\mathcal{L}}(w^{t+1}) \;\le\; \hat{\mathcal{L}}(w^t) - (\eta - kh), \tag{51}$$

and convergence occurs in at most $T \le \lfloor B/(\eta - kh) \rfloor$ iterations.

Proof. Since $\hat{w} = w^t$ with the k bits $\mathcal{I} = \{j_1, \dots, j_k\}$ flipped and $i^* \notin \mathcal{I}$, we telescope BV_{i^*} from $\hat{w}$ back to w^t. Define $u^0 = \hat{w}$ and $u^l = (u^{l-1})^{(j_l)}$ (restoring bit j_l), so $u^k = w^t$. For each l, since $i^* \neq j_l$:

$$\begin{aligned}\mathrm{BV}_{i^*}(u^l) - \mathrm{BV}_{i^*}(u^{l-1}) &= \big[\hat{\mathcal{L}}((u^l)^{(i^*)}) - \hat{\mathcal{L}}(u^l)\big] - \big[\hat{\mathcal{L}}((u^{l-1})^{(i^*)}) - \hat{\mathcal{L}}(u^{l-1})\big] \\ &= H_{i^*,j_l}(u^{l-1}).\end{aligned} \tag{52}$$

The second equality holds because $u^l = (u^{l-1})^{(j_l)}$, so $(u^l)^{(i^*)} = (u^{l-1})^{(j_l, i^*)}$, and the four terms match the definition of $H_{i^*,j_l}(u^{l-1})$. Summing over $l = 1, \dots, k$:

$$\mathrm{BV}_{i^*}(w^t) = \mathrm{BV}_{i^*}(\hat{w}) + \sum_{l=1}^{k} H_{i^*,j_l}(u^{l-1}). \tag{53}$$

Each $|H_{i^*,j_l}(u^{l-1})| \le h$, so $\mathrm{BV}_{i^*}(w^t) < -\eta + kh < 0$ when $\eta > kh$. Therefore $\hat{\mathcal{L}}(w^{t+1}) = \hat{\mathcal{L}}(w^t) + \mathrm{BV}_{i^*}(w^t) < \hat{\mathcal{L}}(w^t) - (\eta - kh)$, and the Lyapunov function $\Phi(w^t) = \hat{\mathcal{L}}(w^t)$ decreases by at least $\eta - kh > 0$ at each accepted step. Since $\Phi \in [0, B]$, at most $\lfloor B/(\eta - kh) \rfloor$ decreases are possible. □

Remark H.4 (The BOLD accumulator as an adaptive threshold). Theorem 5.7 requires a threshold $\eta > 0$ such that bits are flipped only when the base-point signal exceeds η. The BOLD accumulator [42] used in our practical implementation (Remark 4.3) plays an analogous role. It maintains a per-bit state a_i^t that integrates the adversarial signed variation across iterations and flips bit i only when $a_i^t \cdot w_i^t \ge 1$, which is equivalent to requiring the current signal to exceed an effective threshold $\eta_{\text{eff}}(i,t) > 0$ that the plasticity ratio raises in high-flux regions and lowers near convergence. The thresholded quantity is the learning-rate-scaled accumulated backward signal, and the threshold adapts over training. § I derives η_{eff}, proves its positivity, and analyzes its behavior across training phases.

Note that the accumulator-based variant may flip multiple bits simultaneously per step, introducing non-linear interactions (positive Hessian entries H_{ij}). The following proposition quantifies when simultaneous flips are safe.

Proposition H.5 (Multi-Bit Descent under Bounded Hessian). *Under the bounded-Hessian assumption $|H_{ij}(w)| \le h$, suppose the accumulator simultaneously flips a set $S = \{i_1, \dots, i_m\} \subseteq [n]$ of bits in w^t, each satisfying $\mathrm{BV}_{i_l}(w^t) < -\delta$ for some $\delta > 0$. If $\delta > \frac{(m-1)h}{2}$, then the joint flip decreases the empirical loss:*

$$\hat{\mathcal{L}}(w^t \oplus \mathbf{1}_S) - \hat{\mathcal{L}}(w^t) \;\le\; -m\delta + \tbinom{m}{2} h \;<\; 0. \tag{54}$$

Proof. Flip the bits sequentially: define $S_0 = \emptyset$ and $S_l = \{i_1, \dots, i_l\}$. The l-th marginal gain is

$$f(S_l) - f(S_{l-1}) = \mathrm{BV}_{i_l}(w^t \oplus \mathbf{1}_{S_{l-1}}). \tag{55}$$

Using the telescoping identity (the same argument as Theorem 5.8, applied to each earlier flip $j \in S_{l-1}$):

$$\mathrm{BV}_{i_l}(w^t \oplus \mathbf{1}_{S_{l-1}}) = \mathrm{BV}_{i_l}(w^t) + \sum_{j \in S_{l-1}} H_{i_l,j}(\cdot) \le \mathrm{BV}_{i_l}(w^t) + (l-1)h. \tag{56}$$

Summing over $l = 1, \dots, m$ and using $\sum_{l=1}^{m}(l-1) = \binom{m}{2}$:

$$f(S) = \sum_{l=1}^{m} \big[f(S_l) - f(S_{l-1})\big] \le \sum_{l=1}^{m} \big[\mathrm{BV}_{i_l}(w^t) + (l-1)h\big] < -m\delta + \tbinom{m}{2} h. \qquad \square$$

Remark H.6 (Convergence of the accumulator variant). Combining Theorem 5.8 (which gives each accepted bit $\mathrm{BV}_i(w^t) < -(\eta - kh)$) with Proposition 5.11 (setting $\delta = \eta - kh$), the accumulator variant with m simultaneous flips per step converges provided

$$\eta > \left(k + \tfrac{m-1}{2}\right)h. \tag{57}$$

This shows a three-way interaction between the Hamming radius k (adversarial probe strength), the number of simultaneous flips m (optimizer aggressiveness), and the Hessian bound h (landscape curvature). For typical BOLD training where $m \ll n$ and h is moderate, this is a mild strengthening of the single-bit condition $\eta > kh$. When the accumulator flips many bits (m large) or the landscape has strong interactions (h large), the condition requires either a larger threshold η or a smaller Hamming radius k. The condition applies to multi-bit steps whose flipped bits satisfy the premise of Proposition 5.11.

I The BOLD Accumulator as an Adaptive Threshold

This appendix provides a detailed analysis of how the BOLD accumulator [42] used in our practical implementation (Remark 4.3) gives rise to an implicit, adaptive threshold that plays the role of the fixed threshold in Theorem 5.7, applied to the accumulated backward signal rather than to the exact Boolean Variation. We show that the accumulator dynamics induce a per-bit, per-step effective threshold $\eta_{\text{eff}}(i,t) > 0$ that automatically adjusts its sensitivity over the course of training.

I.1 Accumulator Dynamics

The BOLD accumulator maintains, for each bit $i \in [n]$, a scalar state $a_i^t \in \mathbb{R}$ and a plasticity ratio $r_i^t \in [0,1]$. At each Hamming-SAM iteration t, after Phase 1 produces the adversarial point $\hat{w}^t$ a backward pass at $\hat{w}^t$ yields the signal g_i^t, an estimate of $f_i'(\hat{w}^t)$, which does not depend on the current value of bit i (§ B.4). For $i \notin \mathcal{I}$, $g_i^t\, w_i^t$ estimates $-\mathrm{BV}_i(\hat{w}^t)$. For an ascent bit, it estimates $-\mathrm{BV}_i(w^t \oplus \mathbf{1}_{\mathcal{I}\setminus\{i\}}) \le -\mathrm{BV}_i(w^t) + (k-1)h$. The accumulator updates as:

Step 1: Accumulation.

$$a_i^{t+1} = r_i^t \cdot a_i^t + \eta_{\text{lr}} \cdot g_i^t, \tag{58}$$

where $\eta_{\text{lr}} > 0$ is the learning rate.

Step 2: Flip decision. Define the flip indicator set:

$$\mathcal{F}^{t+1} = \left\{ i \in [n] : a_i^{t+1} \cdot w_i^t \ge 1 \right\}. \tag{59}$$

For each $i \in \mathcal{F}^{t+1}$, the weight is flipped: $w_i^{t+1} = -w_i^t$. For $i \notin \mathcal{F}^{t+1}$, $w_i^{t+1} = w_i^t$.

Step 3: Reset and plasticity. After flipping:

$$a_i^{t+1} \leftarrow 0 \quad \text{for all } i \in \mathcal{F}^{t+1}, \tag{60}$$

$$r_i^{t+1} = 1 - \bar{f}^{t+1}, \quad \text{where } \bar{f}^{t+1} = \frac{|\mathcal{F}^{t+1} \cap \mathcal{G}_i|}{|\mathcal{G}_i|} \tag{61}$$

is the fraction of bits in the same parameter tensor $\mathcal{G}_i$ that flipped at step $t+1$.

I.2 Derivation of the Effective Threshold

We now derive the effective threshold $\eta_{\text{eff}}(i,t)$ by analyzing the flip condition (59).

Proposition I.1 (Effective Threshold). *At iteration t, bit i is flipped if and only if the signal $g_i^t \cdot w_i^t$ satisfies*

$$g_i^t \cdot w_i^t \;\ge\; \underbrace{\frac{1 - r_i^t \cdot a_i^t \cdot w_i^t}{\eta_{\text{lr}}}}_{\triangleq\ \eta_{\text{eff}}(i,t)}. \tag{62}$$

Moreover, $\eta_{\text{eff}}(i,t) > 0$ for all i and t.

Proof. Substituting (58) into (59):

$$\begin{aligned} a_i^{t+1} \cdot w_i^t &\geq 1 \\ \left(r_i^t \cdot a_i^t + \eta_{\text{lr}} \cdot g_i^t\right) \cdot w_i^t &\geq 1 \\ r_i^t \cdot a_i^t \cdot w_i^t + \eta_{\text{lr}} \cdot g_i^t \cdot w_i^t &\geq 1 \\ g_i^t \cdot w_i^t &\geq \frac{1 - r_i^t \cdot a_i^t \cdot w_i^t}{\eta_{\text{lr}}}. \end{aligned} \tag{63}$$

For strict positivity of η_{eff}, we need $r_i^t \cdot a_i^t \cdot w_i^t < 1$. This holds by a simple induction argument:

Base case. At initialization, $a_i^0 = 0$ for all i, so $r_i^0 \cdot a_i^0 \cdot w_i^0 = 0 < 1$.

Inductive step. Suppose $r_i^t \cdot a_i^t \cdot w_i^t < 1$ at the start of iteration t. Two cases arise:

- If $i \in \mathcal{F}^{t+1}$ (bit i flips): the accumulator resets to $a_i^{t+1} = 0$ by (60), so $r_i^{t+1} \cdot a_i^{t+1} \cdot w_i^{t+1} = 0 < 1$.
- If $i \notin \mathcal{F}^{t+1}$ (bit i does not flip): by (59), we have $a_i^{t+1} \cdot w_i^t < 1$. Since $w_i^{t+1} = w_i^t$ and $r_i^{t+1} \in [0, 1]$, two subcases arise. If $a_i^{t+1} \cdot w_i^t \geq 0$, then $r_i^{t+1} \cdot a_i^{t+1} \cdot w_i^{t+1} \leq a_i^{t+1} \cdot w_i^t < 1$. If $a_i^{t+1} \cdot w_i^t < 0$, then $r_i^{t+1} \cdot a_i^{t+1} \cdot w_i^{t+1} \leq 0 < 1$.

In both cases the invariant is maintained, so $\eta_{\text{eff}}(i, t) > 0$ for all i, t. □

I.3 Adaptive Behavior Across Training Phases

The plasticity ratio r_i^t adjusts η_{eff} according to the training phase. We analyze three regimes.

Early training (high flip rate). When many bits flip per step, $\bar{f}_{\mathcal{G}_i}^t$ is large and $r_i^t = 1 - \bar{f}_{\mathcal{G}_i}^t$ is small. Consequently, the carry-over term $r_i^t \cdot a_i^t \cdot w_i^t$ is heavily damped, and:

$$\eta_{\text{eff}}(i, t) \approx \frac{1 - 0}{\eta_{\text{lr}}} = \frac{1}{\eta_{\text{lr}}}. \tag{64}$$

The threshold is at its *maximum*. The optimizer requires a strong single-step signal before flipping any bit. This prevents oscillation during the initial phase when the loss landscape is noisy and the sharpness B_k is large.

Mid training (moderate flip rate). As training progresses, fewer bits flip and r_i^t increases toward 1. The accumulator begins to retain history across steps: a_i^t builds up over $\sim 1/(1 - r_i^t)$ iterations. If the signal consistently favors flipping bit i, that is, $g_i^t \cdot w_i^t > 0$ across multiple steps, the accumulated product $r_i^t \cdot a_i^t \cdot w_i^t$ grows, and $\eta_{\text{eff}}(i, t)$ *decreases*. The optimizer becomes progressively more willing to act on weaker but persistent sharpness signals.

Near convergence (low flip rate). When $\bar{f}_{\mathcal{G}_i}^t \approx 0$, we have $r_i^t \approx 1$ and the accumulator is nearly a running sum. After T consistent steps with average signal $\bar{g}_i$:

$$a_i^t \cdot w_i^t \approx T \cdot \eta_{\text{lr}} \cdot \bar{g}_i \cdot w_i^t, \tag{65}$$

so $\eta_{\text{eff}}(i, t) \approx (1 - T\,\eta_{\text{lr}}\,\bar{g}_i \cdot w_i^t)/\eta_{\text{lr}} \to 0^+$ as T grows. The optimizer can therefore detect very small sharpness improvements, which a fixed η would not allow.

I.4 Formal Convergence Guarantee

Theorem I.2 (Properties of Accumulator-Based Hamming-SAM). *Let $\hat{\mathcal{L}} : \{-1,+1\}^n \to [0,B]$ be bounded, and suppose Algorithm 1 with Phase 2 replaced by the accumulator update (58)–(61). Assume the bounded-Hessian condition $|H_{ij}(w)| \le h$ for all $w, i \neq j$. Part (i) holds unconditionally, and parts (ii) and (iii) hold when every flipped bit satisfies $\mathrm{BV}_i(w^t) < -\delta$. Then:*

(i) *Every bit that flips satisfies $g^t_{i^*} \cdot w^t_{i^*} \ge \eta_{\text{eff}}(i^*, t) > 0$ (Proposition I.1), so each accepted flip is justified by strictly positive accumulated evidence.*

(ii) *If at step t the accumulator flips a set $\mathcal{F}^{t+1}$ of m bits, each with $\mathrm{BV}_i(w^t) < -\delta$ for $\delta > 0$, and $\delta > \frac{(m-1)h}{2}$, then $\hat{\mathcal{L}}(w^{t+1}) < \hat{\mathcal{L}}(w^t)$ (Proposition 5.11).*

(iii) *The total number of flips across all bits and all iterations is finite. Specifically, if the conditions of (ii) hold at every flip step with per-flip decrease at least $\delta_{\min} > 0$, then the total number of flip events is at most $\lfloor B/\delta_{\min} \rfloor$.*

Proof.

Part (i). By Proposition I.1, $\eta_{\text{eff}}(i,t) > 0$ for all i, t (the invariant $r^t_i a^t_i w^t_i < 1$ is maintained by the reset mechanism). Therefore every flip is justified by accumulated backward signal exceeding a positive threshold.

Part (ii). This follows directly from Proposition 5.11. The m bits flipped simultaneously each have $\mathrm{BV}_i(w^t) < -\delta$, and the bounded-Hessian assumption controls the interaction cost at $\binom{m}{2}h$. When $\delta > (m-1)h/2$, the joint loss change $\hat{\mathcal{L}}(w^{t+1}) - \hat{\mathcal{L}}(w^t) \le -m\delta + \binom{m}{2}h < 0$.

When the SAM-style evaluation is used (Boolean Variation computed at $\hat{w}$ rather than w^t), Theorem 5.8 gives $\mathrm{BV}_i(w^t) < -(\eta - kh)$ for each accepted bit, so $\delta = \eta - kh$ and the multi-bit condition becomes $\eta > (k + \frac{m-1}{2})h$.

Part (iii). Each flip step that satisfies the condition in (ii) decreases $\hat{\mathcal{L}}(w^t)$ by at least $\delta_{\min} > 0$. Since $\hat{\mathcal{L}} \in [0, B]$, the total number of such steps is at most $\lfloor B/\delta_{\min} \rfloor$. The accumulator reset ensures no bit flips indefinitely: after each flip, the threshold returns to $1/\eta_{\text{lr}}$, requiring fresh evidence for any subsequent flip of the same bit. □

Remark I.3 (Comparison with fixed-η convergence). Theorem 5.7 guarantees convergence of $\hat{\mathcal{L}}(w^t)$ in $\lfloor B/\eta \rfloor$ steps with a fixed threshold η. Theorem 5.8 extends this to the SAM-style algorithm under $\eta > kh$. Theorem I.2 further extends to the multi-bit accumulator under $\eta > (k + \frac{m-1}{2})h$. The three results form a hierarchy, from base-point descent (simplest, strongest bound) → SAM-style single-bit (uses adversarial information, requires bounded Hessian) → accumulator multi-bit (closest to the practical algorithm, under the premise of part (ii)). The decrease guarantee of part (ii) applies at a flip step whose instantaneous signal exceeds $(k + \frac{m-1}{2})h$. Because $\eta_{\text{eff}}(i,t)$ can fall below this level near convergence, flips triggered by accumulated evidence alone are not covered by it. Ascent bits do not satisfy the premise $\mathrm{BV}_i(w^t) < -\delta$, and they receive a non-flip signal whenever $\mathrm{BV}_i(w^t) > (k-1)h$ (§ I.1).

J Detailed Proof of Generalization Bound with Data-Dependent Priors

We derive the PAC-Bayes generalization bound of § 5.2 using data-dependent priors [45, 17]. Split the training data as $\mathcal{D} = \mathcal{S}_0 \sqcup \mathcal{S}_1$. Train a 1,9xreference configuration w_0 on $\mathcal{S}_0$ and define the prior as a product distribution that independently flips each bit of w_0 with probability $p \in (0, \frac{1}{2})$: 1

$$P(w) = \prod_{i=1}^{n} P_i(w_i), \quad P_i(w_i) = \begin{cases} 1-p & \text{if } w_i = (w_0)_i, \\ p & \text{if } w_i \neq (w_0)_i. \end{cases} 1 \tag{66}$$

The posterior Q is the degenerate (deterministic) distribution concentrating on the Hamming-SAM solution $\hat{w}$, which is a valid choice in the PAC-Bayes framework [7]. Because Q is a point mass, $\mathbb{E}_Q[\mathcal{L}] = \mathcal{L}(\hat{w})$, so the bound below controls the loss of the returned configuration itself.

J.1 KL Divergence Computation

Lemma J.1 (KL for Bernoulli Bit-Flip Prior). *Let $d = d_H(\hat{w}, w_0)$ denote the Hamming distance between $\hat{w}$ and the prior center w_0. The KL divergence between the deterministic posterior $Q = \delta_{\hat{w}}$ and the prior P in (66) is:*

$$\mathrm{KL}(Q\|P) = d \cdot \log \frac{1}{p} + (n-d) \cdot \log \frac{1}{1-p}. \tag{67}$$

Proof. Since $Q = \delta_{\hat{w}}$ is a point mass:

$$\mathrm{KL}(Q\|P) = \log \frac{1}{P(\hat{w})} = -\sum_{i=1}^{n} \log P_i(\hat{w}_i). \tag{68}$$

Partition the coordinates into $\mathcal{A} = \{i : \hat{w}_i = (w_0)_i\}$ (agreement, $|\mathcal{A}| = n - d$) and $\mathcal{F} = \{i : \hat{w}_i \neq (w_0)_i\}$ (flipped, $|\mathcal{F}| = d$). Then:

$$\mathrm{KL}(Q\|P) = -\sum_{i\in\mathcal{A}} \log(1-p) - \sum_{i\in\mathcal{F}} \log p = (n-d)\log\frac{1}{1-p} + d\log\frac{1}{p}. \tag{69}$$

□

Optimization over p. Taking the derivative and setting it to zero:

$$\frac{\partial}{\partial p}\mathrm{KL}(Q\|P) = -\frac{d}{p} + \frac{n-d}{1-p} = 0 \quad \Longrightarrow \quad p^* = \frac{d}{n}.$$

At this optimum, $\mathrm{KL}(Q\|P) = n \cdot h(d/n)$, where h is the binary entropy. For $d \ll n$, this scales as $d\log(n/d) + O(d)$, which is much smaller than $n \log 2$.

J.2 Generalization Bound

Theorem J.2 (PAC-Bayes Bound with Data-Dependent Bernoulli Prior). *Let the loss take values in $[0, 1]$, such as the 0-1 classification error. Let w_0 be a configuration trained on $\mathcal{S}_0$, and P be the bit-flip prior (66) centered at w_0 with flip probability $p \in (0, \frac{1}{2})$. Let $\hat{w}$ be the Hamming-SAM solution trained on $\mathcal{S}_1$, with $d = d_H(\hat{w}, w_0)$. Then for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the draw of $\mathcal{S}_1$:*

$$\mathcal{L}(\hat{w}) \;\leq\; \hat{\mathcal{L}}_{\mathcal{S}_1}(\hat{w}) + \sqrt{\frac{d\,\log\frac{1}{p} + (n-d)\,\log\frac{1}{1-p} + \log\frac{2\sqrt{m}}{\delta}}{2m}}, \tag{70}$$

where $m = |\mathcal{S}_1|$.

Proof. We apply the PAC-Bayes–KL inequality of [7] in its McAllester form [40]. For any prior P (independent of $\mathcal{S}_1$) and posterior Q,

$$\mathrm{KL}_{\mathrm{Ber}}\big(\mathbb{E}_Q[\hat{\mathcal{L}}] \,\big\|\, \mathbb{E}_Q[\mathcal{L}]\big) \leq \frac{\mathrm{KL}(Q\|P) + \log(2\sqrt{m}/\delta)}{m}.$$

Applying Pinsker's inequality to the left-hand side and substituting the KL from Lemma 5.5 yields (70), using $\mathbb{E}_Q[\mathcal{L}] = \mathcal{L}(\hat{w})$ for the point-mass posterior. The prior P is constructed from $\mathcal{S}_0$ and is therefore independent of $\mathcal{S}_1$, satisfying the measurability condition. □

Remark J.3 (Optimizing the flip probability p). The bound (70) holds for any fixed $p \in (0, \frac{1}{2})$ chosen independently of $\mathcal{S}_1$. To optimize p post hoc (after observing $d = d_H(\hat{w}, w_0)$, which depends on $\mathcal{S}_1$), we apply a union bound over a dyadic grid $\mathcal{P} = \{2^{-1}, 2^{-2}, \ldots, 2^{-\lceil \log_2 n \rceil}\}$ of $|\mathcal{P}| = \lceil \log_2 n \rceil$ candidate values. Replacing δ by $\delta/|\mathcal{P}|$ in the bound and selecting $p^* = \arg\min_{p\in\mathcal{P}} \mathrm{KL}(Q\|P)$ yields:

$$\mathcal{L}(\hat{w}) \;\leq\; \hat{\mathcal{L}}_{\mathcal{S}_1}(\hat{w}) \;+\; \sqrt{\frac{\mathrm{KL}^* + \log(2\sqrt{m}\,\lceil\log_2 n\rceil\,/\,\delta)}{2m}}, \tag{71}$$

Table 4: PAC-Bayes bounds for 3-layer MLP on CIFAR-10 ($n = 442{,}368$, $m = 25{,}000$, $\delta = 0.05$). Each method uses its own prior trained on $\mathcal{S}_0$, and fine-tuning is on $\mathcal{S}_1$. KL is the minimum over the dyadic grid of Remark J.3. Non-vacuous bounds (< 1.0) are **bolded**.

Method	Epoch	Acc on $\mathcal{S}_1$	d	d/n (%)	KL	Bound
BOLD	1	48.0%	2	0.00	27	**0.548**
	10	47.6%	2,161	0.49	13,706	1.048
	20	48.9%	3,213	0.73	19,034	1.128
	30	49.4%	3,902	0.88	22,372	1.175
	40	49.4%	4,186	0.95	23,747	1.195
	50	49.6%	4,233	0.96	23,975	1.197
Hamming-SAM (k=500)	1	52.1%	0	0.00	1	**0.495**
	10	50.9%	385	0.09	3,101	**0.740**
	20	51.2%	490	0.11	3,828	**0.765**
	30	51.4%	581	0.13	4,459	**0.785**
	40	51.2%	633	0.14	4,813	**0.799**
	50	51.3%	644	0.15	4,881	**0.800**

where $\mathrm{KL}^* = \min_{p \in \mathcal{P}}[d \log(1/p) + (n-d)\log(1/(1-p))]$. The additional penalty $\log\lceil \log_2 n \rceil$ is negligible. For $n = 442{,}368$ it adds $\log 19 \approx 2.9$ to the numerator, which is below 0.1% of the KL term for every bound with $d > 0$ beyond the first epoch, and changes no reported bound by more than 0.001 at epoch 1. In practice, the grid optimum closely approximates the continuous optimum $p^* = d/n$. This is standard practice in PAC-Bayes evaluation [16, 17].

J.3 Empirical Evaluation

We evaluate the bound using the data-split protocol described in § 5.2. CIFAR-10 is split into $\mathcal{S}_0$ (first 25K training samples) and $\mathcal{S}_1$ (last 25K). Each method trains a prior w_0 on $\mathcal{S}_0$, then fine-tunes on $\mathcal{S}_1$ starting from w_0. Each method uses its own prior. Hamming-SAM fine-tunes from a Hamming-SAM prior, and standard BOLD fine-tunes from a BOLD prior. The Hamming distance $d = d_H(\hat{w}, w_0)$ and the bound are computed at each epoch of fine-tuning, with the 0-1 error on $\mathcal{S}_1$ as the empirical risk. Following Remark J.3, we optimize p over a dyadic grid and apply the union-bound-corrected bound (71). The correction adds $\log\lceil \log_2 n \rceil \approx 2.9$ to the numerator, which is negligible relative to the KL term for every bound beyond the first epoch and does not affect non-vacuousness of any reported bound.

Results. Table 4 reports the PAC-Bayes bound for the 3-layer MLP ($n = 442{,}368$, $m = 25{,}000$, $\delta = 0.05$). Two findings stand out.

First, Hamming-SAM achieves non-vacuous bounds at every epoch, while standard BOLD is vacuous after epoch 1. At epoch 50, Hamming-SAM yields a bound of 0.800 with only $d = 644$ bits changed ($d/n = 0.15\%$), whereas BOLD requires $d = 4{,}233$ bits ($d/n = 0.96\%$) and produces a vacuous bound of 1.197. Hamming-SAM flips $6.6\times$ fewer bits during adaptation, which shows that flat basins are more stable under the distribution shift from $\mathcal{S}_0$ to $\mathcal{S}_1$.

Second, the Hamming-SAM prior generalizes well without fine-tuning. At epoch 1 ($d = 0$, before any adaptation), the Hamming-SAM prior already achieves 52.1% accuracy on $\mathcal{S}_1$ versus 48.0% for the BOLD prior, an advantage of 4.1 points from the flat basin alone. The tightest bound (0.495) occurs at this point, reflecting that the Hamming-SAM solution found on $\mathcal{S}_0$ transfers directly to $\mathcal{S}_1$ with no modification.

Scaling behavior. For small p, the dominant KL term is $d \cdot \log(1/p)$, since $\log(1/(1-p)) \approx p$ is negligible. Near the optimum $p \approx d/n$, the bound scales as $O(d \log(n/d))$, so the complexity depends on the *adaptation distance* d rather than the total number of parameters n. For example, for the Hamming-SAM result at epoch 50, the grid optimum $p = 2^{-9}$ gives $\mathrm{KL} \approx 644 \cdot \log 512 + 441{,}724 \cdot \log(512/511) \approx 4{,}881$, and a generalization gap of $\sqrt{(4{,}881 + 11.7)/50{,}000} \approx 0.31$. The uniform prior would give $\mathrm{KL} = 442{,}368 \cdot \log 2 \approx 306{,}600$, yielding a vacuous bound exceeding 2.4.

The results confirm the mechanism described in § 5.2: lower sharpness B_k yields smaller adaptation distance d, which in turn yields a tighter bound. The Hamming-SAM prior, having converged to a flat basin on $\mathcal{S}_0$, is stable under the distribution shift to $\mathcal{S}_1$, and the optimizer needs only 644 bit-flips (0.15% of parameters) to adapt. The BOLD prior lies in a sharper basin and requires $4{,}233$ bit-flips (0.96%) to reach a reasonable solution on $\mathcal{S}_1$, and even then the bound remains vacuous.

K Forman-Ricci Curvature: Full Analysis

The preceding sections established that Hamming-SAM finds weight configurations with uniformly low loss across a Hamming ball. In this appendix, we ask a structural question: *what geometric property of the loss landscape distinguishes a truly flat basin from a narrow valley that only has low worst-case loss?* In continuous optimization, this question is answered by Ricci curvature, which measures whether neighboring geodesics converge (positive curvature, robust basins) or diverge (negative curvature, brittle saddle regions). We develop the discrete analogue for the Boolean hypercube using Forman-Ricci curvature [20], providing a combinatorial characterization of flatness that complements the algebraic (PAC-Bayes) and combinatorial (compression) perspectives of § 4.

K.1 The Loss-Weighted Hamming Graph

The key construction is to embed the loss landscape into the geometry of the Hamming graph by assigning weights that reflect the local loss structure.

***Definition* K.1** (Loss-Weighted Hamming Graph)**.** *Given a Boolean weight space $\{-1,+1\}^n$ and empirical risk $\hat{\mathcal{L}}$, define the loss-weighted Hamming graph $G_\beta = (V, E, \omega)$ with:*

- Node weights: $\omega(w) = \exp(-\beta\,\hat{\mathcal{L}}(w))$ *for a temperature $\beta > 0$, concentrating mass on low-loss configurations.*
- Edge weights: $\omega(w, w^{(i)}) = \exp(-\beta\,|\mathrm{BV}_i(w)|)$*, weighting edges by the stability of the corresponding bit-flip. Stable flips with $|\mathrm{BV}_i| \approx 0$ receive weight close to 1, while disruptive flips receive low weight.*

The parameter β controls the sensitivity of the geometry to loss differences. In the limit $\beta \to 0$, G_β reduces to the unweighted Hamming graph. As $\beta \to \infty$, the geometry concentrates entirely on global minima.

K.2 Forman-Ricci Curvature on Weighted Graphs

Forman [20] introduced a combinatorial analogue of Ricci curvature for CW-complexes that depends only on local edge–node–edge adjacency. Specialized to our weighted graph:

***Definition* K.2** (Forman-Ricci Curvature [20])**.** *For an edge $e = (w, w^{(j)})$ in the loss-weighted Hamming graph G_β:*

$$\mathrm{Ric}_F(e) = \omega(e)\left[\frac{\omega(w)+\omega(w^{(j)})}{\omega(e)} - \sum_{\substack{e'\sim w\\ e'\neq e}} \frac{\omega(w)}{\sqrt{\omega(e)\,\omega(e')}} - \sum_{\substack{e'\sim w^{(j)}\\ e'\neq e}} \frac{\omega(w^{(j)})}{\sqrt{\omega(e)\,\omega(e')}}\right], \tag{72}$$

where the sums range over edges incident to w and $w^{(j)}$, respectively, and each vertex has exactly $n-1$ other incident edges.

The following lemma simplifies the curvature formula for the specific structure of the loss-weighted Hamming graph, making the dependence on the loss landscape explicit.

Lemma K.3 (Simplified Forman-Ricci on the Loss-Weighted Hamming Graph). *Let $e_j = (w, w^{(j)})$ be an edge with weight $\omega_j \triangleq \omega(e_j) = \exp(-\beta|\mathrm{BV}_j(w)|)$. Then:*

$$\mathrm{Ric}_F(e_j) = \omega(w) + \omega(w^{(j)}) - \sum_{\substack{i=1 \\ i\neq j}}^{n} \left[\omega(w)\sqrt{\frac{\omega_j}{\omega_i}} + \omega(w^{(j)})\sqrt{\frac{\omega_j}{\omega_i'}}\right], \tag{73}$$

where $\omega_i = \exp(-\beta|\mathrm{BV}_i(w)|)$ and $\omega_i' = \exp(-\beta|\mathrm{BV}_i(w^{(j)})|)$ are the edge weights at w and $w^{(j)}$, respectively.

Proof. The n-dimensional hypercube is n-regular, so each vertex has exactly n incident edges. Expanding (72), the first term is $\omega(e) \cdot [\omega(w) + \omega(w^{(j)})]/\omega(e) = \omega(w) + \omega(w^{(j)})$. Each of the two sums ranges over $n-1$ edges (all neighbors except the edge e_j itself). Distributing the $\omega(e)$ into the sums, the terms become $\omega(w)\sqrt{\omega(e)/\omega(e')}$. Substituting the specific edge weights $\omega(e') = \exp(-\beta|\mathrm{BV}_i(w)|) = \omega_i$ for edges $e' \sim w$ and $\omega(e') = \omega_i'$ for edges $e' \sim w^{(j)}$ yields (73). □

The structure of (73) shows a competition. The positive term rewards nodes with low loss (high ω), while the negative sum penalizes disparity between the edge e_j and its neighboring edges. Curvature is high when both endpoint losses and all neighboring Boolean Variations are approximately uniform, which is the signature of a flat basin.

K.3 Curvature and Sharpness

The following bounds relate the Forman-Ricci curvature to the sharpness B_k. They provide a computable diagnostic of local flatness.

Proposition K.4 (Curvature Bounds under Low Sharpness). *Let w^* be a local minimum of the Hamming-SAM objective $\mathcal{L}_{\mathrm{adv}}(w) = \max_{w' \in \mathcal{B}_k(w)} \hat{\mathcal{L}}(w')$, and define the sharpness at w^* as $B_k \triangleq \mathcal{L}_{\mathrm{adv}}(w^*) - \hat{\mathcal{L}}(w^*)$. Then for any edge $e_j = (w^*, w^{*,(j)})$ with $j \in [n]$:*

(i) ***(Node weight concentration)*** *The node weight ratio satisfies $\omega(w^{*,(j)})/\omega(w^*) \geq \exp(-\beta B_k)$.*

(ii) ***(Flat-basin limit)*** *If in addition w^* is a local minimum of $\hat{\mathcal{L}}$ and $|H_{ij}(w^*)| \leq h$ for all $i \neq j$, then as $B_k, h \to 0$ all edge curvatures $\mathrm{Ric}_F(e_j)$ converge to $\mathrm{Ric}_F^{\mathrm{unif}} = 2\omega^* - 2(n-1)\omega^* = 2\omega^*(1-(n-1))$, the curvature of the unweighted hypercube rescaled by the node weight $\omega^* = \exp(-\beta\hat{\mathcal{L}}(w^*))$.*

Proof.

Part (i). Since $w^{*,(j)} \in \mathcal{B}_1(w^*) \subseteq \mathcal{B}_k(w^*)$ for $k \geq 1$, the definition of the sharpness gives $\hat{\mathcal{L}}(w^{*,(j)}) \leq \mathcal{L}_{\mathrm{adv}}(w^*) = \hat{\mathcal{L}}(w^*) + B_k$. Therefore:

$$\begin{aligned}\frac{\omega(w^{*,(j)})}{\omega(w^*)} &= \exp\big(-\beta[\hat{\mathcal{L}}(w^{*,(j)}) - \hat{\mathcal{L}}(w^*)]\big) \\ &\geq \exp(-\beta B_k).\end{aligned} \tag{74}$$

Part (ii). At a local minimum, $0 \leq \mathrm{BV}_i(w^*) \leq B_1(w^*) \leq B_k$, and $\mathrm{BV}_i(w^{*,(j)}) = \mathrm{BV}_i(w^*) + H_{ij}(w^*)$, so $|\mathrm{BV}_i(w^{*,(j)})| \leq B_k + h$. As $B_k, h \to 0$, all edge weights ω_i, ω_i' tend to 1 and, by part (i), $\omega(w^{*,(j)}) \to \omega(w^*) = \omega^*$. Substituting into Lemma K.3:

$$\mathrm{Ric}_F(e_j) \;\to\; \omega^* + \omega^* - \sum_{i\neq j}\big[\omega^* + \omega^*\big] = 2\omega^* - 2(n-1)\omega^*.$$

For the unweighted case ($\beta = 0$, so $\omega^* = 1$), this reduces to $2 - 2(n-1) < 0$ for $n > 2$, recovering the known negative curvature of the hypercube graph. □

K.4 Average Curvature and Sharpness

The individual-edge result can be aggregated into a statement linking the average curvature to the Hamming-SAM objective.

Theorem K.5 (Average Curvature Lower Bound)**.** *Let w^* be a Hamming-SAM minimum with sharpness B_k and loss $\ell^* = \hat{\mathcal{L}}(w^*)$. Assume further that w^* is a local minimum of $\hat{\mathcal{L}}$, i.e., $\mathrm{BV}_i(w^*) \geq 0$ for all $i \in [n]$, and that $|H_{ij}(w^*)| \leq h$ for all $i \neq j$. Define the average Forman-Ricci curvature over all edges incident to w^*:*

$$\overline{\mathrm{Ric}}_F(w^*) \triangleq \frac{1}{n}\sum_{j=1}^{n} \mathrm{Ric}_F(w^*, w^{*,(j)}).$$

Then:

$$\overline{\mathrm{Ric}}_F(w^*) \;\geq\; e^{-\beta\ell^*}\Big[1 + e^{-\beta B_k} - (n-1)\, e^{\beta B_k/2}\big(1 + e^{\beta h/2}\big)\Big]. \tag{75}$$

In particular, the right-hand side is decreasing in B_k, so minimizing the Hamming-SAM sharpness B_k maximizes the curvature lower bound.

Proof. For each edge $e_j = (w^*, w^{*,(j)})$, the positive term in (73) satisfies $\omega(w^*) + \omega(w^{*,(j)}) \geq e^{-\beta\ell^*}(1 + e^{-\beta B_k})$ by part (i) of Proposition K.4.

For the negative terms, we seek an upper bound. The edge weights satisfy $\omega_j = e^{-\beta|\mathrm{BV}_j|} \leq 1$. For the edges incident to w^*, the local minimum assumption gives $0 \leq \mathrm{BV}_i(w^*) \leq B_1(w^*) \leq B_k$, so $\omega_i \geq e^{-\beta B_k}$ and $\sqrt{\omega_j/\omega_i} \leq e^{\beta B_k/2}$. For the edges incident to the neighbor $w^{*,(j)}$, flipping bit j changes the variation of bit i by one interaction entry, $\mathrm{BV}_i(w^{*,(j)}) = \mathrm{BV}_i(w^*) + H_{ij}(w^*)$, so $|\mathrm{BV}_i(w^{*,(j)})| \leq B_k + h$, which gives $\omega_i' \geq e^{-\beta(B_k+h)}$ and $\sqrt{\omega_j/\omega_i'} \leq e^{\beta(B_k+h)/2}$.

Applying these bounds to the negative sums from Lemma K.3:

$$\begin{aligned}\text{Negative Terms} &\leq \sum_{i\neq j} \omega(w^*)\, e^{\beta B_k/2} + \sum_{i\neq j} \omega(w^{*,(j)})\, e^{\beta(B_k+h)/2} \\ &\leq (n-1)\, e^{-\beta\ell^*}\, e^{\beta B_k/2}\big(1 + e^{\beta h/2}\big),\end{aligned}$$

where the second step uses $\omega(w^{*,(j)}) \leq \omega(w^*) = e^{-\beta\ell^*}$, since w^* is a local minimum of the empirical loss and its neighbor has equal or higher loss.

Subtracting this upper bound from the positive term lower bound yields

$$\mathrm{Ric}_F(e_j) \geq e^{-\beta\ell^*}\Big[1 + e^{-\beta B_k} - (n-1)\, e^{\beta B_k/2}\big(1 + e^{\beta h/2}\big)\Big].$$

The right-hand side does not depend on j, so averaging over all n edges yields (75).

Monotonicity in B_k: As $B_k, h \to 0$, the terms $e^{-\beta B_k}$, $e^{\beta B_k/2}$ and $e^{\beta h/2}$ all tend to 1, and the bound reduces to $e^{-\beta\ell^*}[2 - 2(n-1)]$, exactly recovering the curvature of the unweighted hypercube scaled by the node weight (Proposition K.4(ii)). As B_k decreases, that is, as the basin becomes flatter, the positive term $e^{-\beta B_k}$ grows and the subtracted penalty $e^{\beta B_k/2}$ shrinks, monotonically increasing the curvature lower bound. □

Remark K.6 (On the local minimum assumption). The assumption that w^* is a local minimum of $\hat{\mathcal{L}}$ (not just of $\mathcal{L}_{\mathrm{adv}}$) is used in the proof to ensure $0 \leq \mathrm{BV}_i(w^*) \leq B_k$ and $\omega(w^{*,(j)}) \leq \omega(w^*)$ for all neighbors. This is mild in practice. The convergence analysis (Theorems 5.7 and 5.8) shows that Hamming-SAM drives $\hat{\mathcal{L}}(w^t)$ to an η-stationary point where $\mathrm{BV}_i(w^t) \geq -\eta$ for all i, so the assumption holds approximately at convergence with tolerance η.

Remark K.7 (Interpretation). Theorem K.5 formalizes the intuition that Hamming-SAM converges to regions of the Hamming graph with higher Forman-Ricci curvature. Informally, high curvature on this graph means that the loss and the Boolean Variations vary little across the edges incident to a vertex, which is the combinatorial counterpart of a basin with slow loss variation. In the ablation of § 7.2, curvature is highest at k=10000, which has the flattest neighborhood but the lowest test accuracy, so curvature measures local flatness rather than generalization.

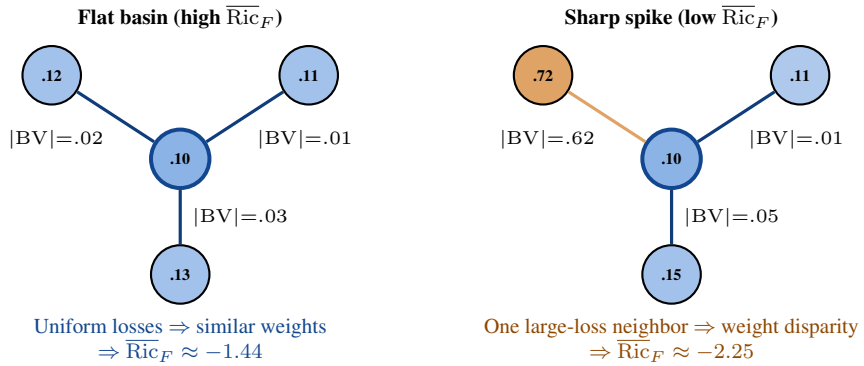


Figure 8: **Forman-Ricci curvature on the loss-weighted Hamming graph** (one vertex and its n=3 neighbors, β=3). *Left:* Flat basin with losses 0.10–0.13. Nearly uniform weights give $\overline{\mathrm{Ric}}_F \approx -1.44$. *Right:* Sharp spike (one neighbor at 0.72). The low-weight edge lowers the curvature of the other two edges, giving $\overline{\mathrm{Ric}}_F \approx -2.25$. Hamming-SAM favors the left configuration.

K.5 Estimating the Average Curvature

Evaluating $\overline{\mathrm{Ric}}_F$ exactly requires all n edges at w and all $n-1$ edges at each neighbor. We estimate it by Monte Carlo, after projecting the weights onto $\{-1,+1\}^n$. We sample 200 edges e_j uniformly, and for each, estimate the two negative sums of Lemma K.3 from the same 50 uniformly sampled bits at w and at $w^{(j)}$, rescaled by $(n-1)/50$ (random seed 42). All node and edge weights use exact finite differences of the cross-entropy on a fixed batch of 256 training images, with $\beta = 1$. The two-hop weights ω_i' are measured directly at $w^{(j)}$, so the estimate does not rely on the bounded-interaction assumption. The values reported in § 7.2 are this average divided by $(n-1)\,\omega(w)$, which removes the dependence on the network size and on the loss level and equals -2 in the flat-basin limit of Proposition K.4.

K.6 Worked Example: Curvature on $\{-1,+1\}^3$

To make the curvature analysis concrete, we compute Ric_F for the two scenarios of Fig. 8, assuming no pairwise interaction, so that the edges incident to $w^{(j)}$ carry the same $|\mathrm{BV}_i|$ as those incident to w. We use $\beta = 3$ so that the contrast is visible with three neighbors. The measurements of § 7.2 use $\beta = 1$ (§ K.5).

Flat basin. The center has loss 0.10 and its neighbors $0.12, 0.11, 0.13$, so $\omega(w) = e^{-0.30} \approx 0.741$. All three edges have similar weight, and Lemma K.3 gives $\mathrm{Ric}_F(e_1,e_2,e_3) \approx (-1.44,-1.53,-1.35)$, an average of $\overline{\mathrm{Ric}}_F \approx -1.44$.

Sharp spike. Replacing the first neighbor by a loss of 0.72 gives the large-variation edge a small weight, $\omega(e_1) = e^{-1.86} \approx 0.156$. Because $\omega(e_1)$ enters the numerator of its own ratios, this edge becomes mildly positively curved ($\mathrm{Ric}_F(e_1) \approx 0.15$). It also appears in the denominator of the ratios of the two other edges, whose curvatures drop to -3.73 and -3.16. The vertex average falls to $\overline{\mathrm{Ric}}_F \approx -2.25$, about 56% more negative than in the flat basin. It is the average over incident edges, the quantity bounded in Theorem K.5 and estimated in § K.5, that separates the two neighborhoods.

K.7 Discussion

Remark K.8 (Novelty). While Forman-Ricci curvature has been applied to data graphs [53], this is, to our knowledge, the first application to a neural network's *parameter space*. The loss-weighted Hamming graph construction (Definition K.1) is original to this work.

Remark K.9 (Relation to other curvatures). Several discrete Ricci curvatures exist. The discrete Bonnet–Myers theorem of [44], which links positive curvature to diameter and random-walk mixing, is stated for Ollivier (coarse) Ricci curvature. The Forman curvature used here is a different notion, which we adopt because it is computable directly from local node and edge weights.

L BDM Regularization: Full Analysis and Empirical Evidence

The theoretical framework developed so far characterizes flatness through worst-case loss (§ 4), graph curvature (§ K), and PAC-Bayes complexity (§ 5.2). In this appendix, we introduce a complementary perspective from algorithmic information theory. *Simpler* weight configurations, those with lower algorithmic complexity, should generalize better, since they encode fewer arbitrary choices and are therefore less likely to overfit. For Boolean networks, this intuition can be made concrete using the Block Decomposition Method (BDM), which approximates Kolmogorov complexity for finite binary strings [57]. BDM is non-differentiable, so gradient-based optimizers cannot use it. The gradient-free optimization of Hamming-SAM allows its direct use as a training-time regularizer, addressing an open problem raised in recent work [51].

L.1 The Block Decomposition Method

***Definition* L.1** (Block Decomposition Method [57])**.** *Let $s \in \{0,1\}^n$ be a binary string (obtained from a Boolean weight vector $w \in \{-1,+1\}^n$ via the map $s_i = (1+w_i)/2$). The BDM partitions s into non-overlapping blocks $b_1, b_2, \ldots, b_M$ of fixed size d (typically $d = 12$), with each block's algorithmic complexity estimated by the Coding Theorem Method (CTM):*

$$\mathrm{CTM}(b) = -\log_2 \mathbb{P}_U(b), \tag{76}$$

where $\mathbb{P}_U(b)$ is the probability that a random Turing machine of bounded size produces b, precomputed and tabulated for all 2^d strings of length d. The BDM aggregates block complexities with a logarithmic correction for repeated patterns:

$$\mathrm{BDM}(s) = \sum_{u \in \mathcal{U}} \big[\mathrm{CTM}(u) + \log_2 n_u\big], \tag{77}$$

where $\mathcal{U}$ is the set of distinct block values occurring in the decomposition and n_u is the multiplicity of block value u. Intuitively, $\mathrm{BDM}(s)$ is small when s consists of few distinct simple blocks repeated many times (high compressibility), and large when s contains many distinct complex blocks (low compressibility).

Remark L.2 (BDM as an approximation to Kolmogorov complexity). While Kolmogorov complexity $K(s)$ is uncomputable, BDM is computable and upper-bounds $K(s)$ up to logarithmic terms. It is in turn bounded above by a block-entropy term, to which it reduces when all blocks look random [57]. Between these two regimes, BDM captures repeated structure that Shannon entropy does not see. This matters for Boolean weight matrices, which develop repeated motifs during training, as observed empirically by [51].

L.2 BDM-Regularized Hamming-SAM Objective

The key observation enabling BDM regularization is that Hamming-SAM operates via discrete bit-flips, not gradients. Since the BDM of a binary string can be evaluated exactly for any configuration, and a single bit-flip changes exactly one block of the decomposition, the *change* in BDM under a bit-flip is efficiently computable.

***Definition* L.3** (BDM-Regularized Objective)**.** *For a regularization parameter $\lambda \geq 0$, define the BDM-regularized Hamming-SAM objective:*

$$\mathcal{L}_{\mathrm{adv}}^{\mathrm{BDM}}(w) \triangleq \mathcal{L}_{\mathrm{adv}}(w) + \lambda \cdot \mathrm{BDM}(w), \tag{78}$$

and the corresponding BDM-augmented Boolean Variation:

$$\mathrm{BV}_i^{\mathrm{BDM}}(w) \triangleq \mathrm{BV}_i(w) + \lambda\big[\mathrm{BDM}(w^{(i)}) - \mathrm{BDM}(w)\big], \tag{79}$$

where $\mathrm{BV}_i(w) = \hat{\mathcal{L}}(w^{(i)}) - \hat{\mathcal{L}}(w)$ is the standard Boolean Variation.

The BDM-augmented Boolean Variation (79) replaces the standard BV in Algorithm 1 without any other modification. Phase 1 selects the at most k bits with largest positive $\mathrm{BV}_i^{\mathrm{BDM}}$ (adversarial probing that also seeks high-complexity perturbations), and Phase 2 flips the bit with most negative $\mathrm{BV}_i^{\mathrm{BDM}}$ at the adversarial point (descent toward both low loss and low complexity).

Proposition L.4 (Efficient BDM Variation). *For a weight vector $w \in \{-1,+1\}^n$ partitioned into non-overlapping blocks of size d, flipping bit i changes exactly one block of the decomposition and therefore at most two entries of the multiplicity table. Consequently:*

(i) $\mathrm{BDM}(w^{(i)}) - \mathrm{BDM}(w)$ *can be computed in $O(1)$ time, given the precomputed CTM table and the current block decomposition.*

(ii) *The full vector of BDM differences $\big(\mathrm{BDM}(w^{(i)}) - \mathrm{BDM}(w)\big)_{i=1}^{n}$ can be computed in $O(n)$ time, so the BDM-augmented Boolean Variation adds only $O(n)$ operations to the backward pass that produces the standard Boolean Variations.*

Proof. Part (i). The BDM partition divides $s \in \{0,1\}^n$ into $M = \lceil n/d \rceil$ non-overlapping blocks, so position i belongs to exactly one block, whose value changes from some u to some u' under the flip. By (77), the only affected terms are those of u and u': the multiplicity n_u decreases by one and $n_{u'}$ increases by one, with CTM values read from the precomputed table. Updating the sum therefore requires a constant number of operations.

Part (ii). Each of the n bits affects one block, so computing all n differences $\mathrm{BDM}(w^{(i)}) - \mathrm{BDM}(w)$ requires $O(n)$ total operations. □

The same argument applies to the 2D spatial variant used in § L.5.2, where each bit belongs to exactly one 3×3 filter block.

L.3 Convergence of BDM-Regularized Hamming-SAM

We now check whether adding the BDM regularizer preserves the convergence guarantees of Theorem 5.7. The following result confirms this for the base-point descent rule.

Proposition L.5 (Convergence with BDM Regularization). *Let $B_{\max} = \max_w \hat{\mathcal{L}}(w)$ and $C_{\max} = \max_w \mathrm{BDM}(w)$. If Algorithm 1 is run with $\mathrm{BV}^{\mathrm{BDM}}$ in place of BV and accepts bit-flips only when $\mathrm{BV}^{\mathrm{BDM}}_{i^*}(w^t) < -\eta$, evaluated at the base point as in Theorem 5.7, for threshold $\eta > 0$, then the algorithm converges to an η-stationary point of $\hat{\mathcal{L}}(w) + \lambda\,\mathrm{BDM}(w)$ in at most*

$$T \;\le\; \left\lfloor \frac{B_{\max} + \lambda\, C_{\max}}{\eta} \right\rfloor \tag{80}$$

iterations.

Proof. Define the Lyapunov function $\Phi^{\mathrm{BDM}}(w^t) = \hat{\mathcal{L}}(w^t) + \lambda\,\mathrm{BDM}(w^t)$. The BDM-augmented Boolean Variation $\mathrm{BV}^{\mathrm{BDM}}_i(w) = \mathrm{BV}_i(w) + \lambda[\mathrm{BDM}(w^{(i)}) - \mathrm{BDM}(w)]$ equals exactly $\Phi^{\mathrm{BDM}}(w^{(i)}) - \Phi^{\mathrm{BDM}}(w)$. Since the accepted flip is applied to w^t and the acceptance test $\mathrm{BV}^{\mathrm{BDM}}_{i^*}(w^t) < -\eta$ is evaluated at w^t, each accepted step gives $\Phi^{\mathrm{BDM}}(w^{t+1}) \le \Phi^{\mathrm{BDM}}(w^t) - \eta$, as in Theorem 5.7. Since $\Phi^{\mathrm{BDM}} \ge 0$ and $\Phi^{\mathrm{BDM}}(w^0) \le B_{\max} + \lambda\, C_{\max}$, at most $\lfloor (B_{\max} + \lambda\, C_{\max})/\eta \rfloor$ accepted steps are possible. □

L.4 Connection to the PAC-Bayes Framework

The BDM regularizer can be understood as an *algorithmic complexity prior* within the PAC-Bayes framework. The following result makes this connection precise.

Proposition L.6 (BDM as a Coding-Length Prior). *Define a prior P_{BDM} over $\{-1,+1\}^n$ by $P_{\mathrm{BDM}}(w) \propto 2^{-\mathrm{BDM}(w)}$, assigning higher probability to algorithmically simpler configurations. Then for the deterministic posterior $Q = \delta_{\hat{w}}$:*

$$\mathrm{KL}(Q \| P_{\mathrm{BDM}}) = \mathrm{BDM}(\hat{w}) \cdot \log 2 + \log Z_{\mathrm{BDM}}, \tag{81}$$

where $Z_{\mathrm{BDM}} = \sum_w 2^{-\mathrm{BDM}(w)}$ is the normalizing constant. For a loss with values in $[0,1]$, the PAC-Bayes bound with this prior is increasing in both $\hat{\mathcal{L}}(\hat{w})$ and $\mathrm{BDM}(\hat{w})$.

Proof. For the deterministic posterior $Q = \delta_{\hat{w}}$:

$$\begin{aligned}\mathrm{KL}(Q\|P_{\mathrm{BDM}}) &= -\log P_{\mathrm{BDM}}(\hat{w}) = -\log \frac{2^{-\mathrm{BDM}(\hat{w})}}{Z_{\mathrm{BDM}}} \\ &= \mathrm{BDM}(\hat{w}) \cdot \log 2 + \log Z_{\mathrm{BDM}}. \end{aligned} \tag{82}$$

The prior does not depend on the data, so substituting into the McAllester PAC-Bayes bound [40] gives

$$\mathcal{L}(\hat{w}) \leq \hat{\mathcal{L}}(\hat{w}) + \sqrt{\frac{\mathrm{BDM}(\hat{w}) \cdot \log 2 + \log Z_{\mathrm{BDM}} + \log(2\sqrt{m}/\delta)}{2m}}.$$

The constant $\log Z_{\mathrm{BDM}}$ does not depend on $\hat{w}$, so the bound depends on the solution only through $\hat{\mathcal{L}}(\hat{w})$ and $\mathrm{BDM}(\hat{w})$, and is increasing in both. □

Minimizing $\hat{\mathcal{L}} + \lambda\,\mathrm{BDM}$ therefore decreases both data-dependent terms of this bound, with λ playing the role of the local exchange rate between them.

Remark L.7 (Why BDM and not Shannon entropy?). One might consider regularizing with the Shannon entropy of the weights, $n\,H_{\mathrm{b}}(\hat{p})$ in bits, where $\hat{p}$ is the fraction of $+1$ entries and H_{b} the binary entropy. However, Shannon entropy measures only first-order symbol frequencies and is maximized by a random bit-string, which is the opposite of what we want. BDM captures *structural* simplicity. A weight matrix with repeated motifs (e.g., convolutional filter symmetries or tiled logic patterns) has low BDM but may have near-maximal Shannon entropy. The learning-as-compression hypothesis [51] specifically predicts that trained BNNs develop such structured motifs, making BDM the appropriate complexity measure.

Remark L.8 (Resolving the differentiability barrier). [51] empirically tracked BDM during BNN training and observed that weight matrices converge toward lower algorithmic complexity, but explicitly concluded that BDM "is inherently non-continuous and non-differentiable," making it "unsuitable as a training-time regularizer." Hamming-SAM removes this barrier. Since it uses bit-flips rather than gradients, the BDM-augmented Boolean Variation (79) provides exact, efficiently computable descent directions for the non-differentiable BDM objective. To our knowledge, this is the first use of algorithmic complexity as a *training-time* regularizer rather than a post-hoc diagnostic.

L.5 Empirical Evaluation

We test whether a non-differentiable complexity penalty can be optimized directly within Hamming-SAM, by augmenting its objective with the BDM of the Boolean weights. The penalty favors configurations built from simpler, repeated structural motifs.

L.5.1 Boolean MLP

Experimental Setup. We first evaluate this regularizer on a Boolean MLP trained on CIFAR-10. We apply a global 1D BDM penalty to the concatenated Boolean weight vectors, controlled by a scaling hyperparameter λ, alongside the neighborhood perturbation for Hamming-SAM ($k = 50$). The empirical results are detailed in Table 5.

Table 5: Algorithmic complexity and generalization metrics for a Boolean MLP on CIFAR-10. BDM represents the global 1D algorithmic complexity approximation, while Shannon entropy (total bits, $n\,H_{\mathrm{b}}(\hat{p})$) tracks first-order bit statistics. Unique block values are measured against the 36,864 blocks of length $d = 12$ in the decomposition, of which at most $2^{12} = 4{,}096$ can be distinct.

Method	λ	Test Acc (%)	BDM Score	Shannon Entropy	Unique Blocks
Standard BOLD	-	51.58 ± 0.12	139,403 ± 14	442,368 ± 5	3,968
Hamming-SAM ($k = 50$)	0	52.11 ± 0.09	138,031 ± 11	442,360 ± 4	3,932
Hamming-SAM + BDM	10^{-4}	52.88 ± 0.07	138,107 ± 9	442,364 ± 3	3,936
Hamming-SAM + BDM	10^{-3}	**52.93** ± 0.05	138,341 ± 12	442,362 ± 4	3,942
Hamming-SAM + BDM	10^{-2}	51.57 ± 0.11	138,691 ± 15	442,365 ± 6	3,950
Hamming-SAM + BDM	10^{-1}	51.27 ± 0.14	138,577 ± 13	442,365 ± 5	3,943

The effect of BDM regularization on generalization. Adding a moderate BDM penalty ($\lambda \in [10^{-4}, 10^{-3}]$) improves test accuracy (peak 52.93% at $\lambda = 10^{-3}$, +0.82 points over unregularized Hamming-SAM). Stronger regularization ($\lambda \geq 10^{-2}$) degrades performance. The final BDM scores remain close to the unregularized value, indicating that the penalty acts mainly on the optimization trajectory by discouraging complex intermediate configurations, much as weight decay can improve generalization without producing the smallest-norm solution.

Flatness implicitly reduces algorithmic complexity. Comparing standard BOLD to unregularized Hamming-SAM ($\lambda = 0$), sharpness-aware training reduces the BDM score from 139,403 to 138,031, about 1%, and the number of unique blocks from 3,968 to 3,932, without any explicit complexity penalty. The effect is small but consistent with flat basins having slightly more regular structure.

The blindness of statistical entropy. Across all configurations in Table 5, the Shannon entropy barely changes. In contrast, the BDM score fluctuates and unique block counts decrease under regularization. This occurs because the global bit-balance of the Boolean weights remains near 50/50. As a statistical metric, Shannon entropy only measures the overall frequency of bits and ignores their arrangement. BDM, however, is sensitive to the arrangement and reuse of local motifs. The static entropy also indicates that the regularizer does not collapse the weights toward a degenerate configuration.

L.5.2 2D Spatial BDM on ResNet-18

The global saturation limit. Applying a global 1D algorithmic complexity penalty to large convolutional networks has a practical limit. The maximum number of distinct 1D blocks of length 12 ($2^{12} = 4{,}096$) is quickly saturated by the large number of parameters. Under global saturation, standard BDM and Shannon entropy become insensitive to structural shifts in the optimization trajectory.

Transitioning to 2D spatial regularization. To resolve this, we move from a global 1D penalty to a local 2D spatial BDM penalty. Rather than flattening the entire network, we regularize the algorithmic complexity of the individual 3×3 convolutional filters during training, encouraging spatially smooth, low-complexity feature extractors. We evaluate this approach on a Boolean ResNet-18 (Table 6).

Table 6: Generalization and global complexity metrics for a Boolean ResNet-18 on CIFAR-10. The Hamming-SAM + BDM models were optimized using a localized 2D spatial BDM penalty on the 3×3 convolutional filters. Global BDM, Shannon entropy (total bits) and unique block values (against the 929,792 blocks of the decomposition) are reported post-training, illustrating the saturation of 1D global metrics in deep architectures.

Method	λ	Test Acc (%)	Test Loss	Global BDM	Shannon Entropy	Unique Blocks
Standard BOLD	-	92.99 $\pm$ 0.11	0.566 $\pm$ 0.012	163,039 $\pm$ 18	11,149,042 $\pm$ 85	4,096
Hamming-SAM ($k = 100$)	0	93.22 $\pm$ 0.08	0.399 $\pm$ 0.007	163,108 $\pm$ 14	11,150,255 $\pm$ 62	4,096
Hamming-SAM + BDM	10^{-5}	93.35 $\pm$ 0.06	0.391 $\pm$ 0.005	163,077 $\pm$ 12	11,150,965 $\pm$ 45	4,096
Hamming-SAM + BDM	10^{-4}	**93.63** $\pm$ 0.04	**0.392** $\pm$ 0.004	163,123 $\pm$ 10	11,150,910 $\pm$ 38	4,096
Hamming-SAM + BDM	10^{-3}	93.42 $\pm$ 0.07	0.442 $\pm$ 0.008	163,148 $\pm$ 15	11,150,529 $\pm$ 55	4,096
Hamming-SAM + BDM	10^{-2}	93.37 $\pm$ 0.09	0.476 $\pm$ 0.010	163,075 $\pm$ 16	11,149,457 $\pm$ 70	4,096
Hamming-SAM + BDM	10^{-1}	93.24 $\pm$ 0.12	0.473 $\pm$ 0.011	163,039 $\pm$ 20	11,149,351 $\pm$ 80	4,096

Effect on generalization. Spatial BDM gives a clear regularization curve. Hamming-SAM alone improves standard training from 92.99% to 93.22%. Adding the spatial penalty improves accuracy further at every λ tested, peaking at 93.63% at $\lambda = 10^{-4}$, a gain of 0.41 points over Hamming-SAM. Test loss improves only for $\lambda \leq 10^{-4}$ and rises above the Hamming-SAM value for larger penalties.

Algorithmic versus statistical complexity at scale. The global evaluation metrics in Table 6 further show the difference between statistical and algorithmic complexity at scale. Across all optimization strategies, the global Shannon entropy remains virtually static, varying by less than 0.02%. Because the network maintains a highly balanced ratio of $+1$ and -1 weights, statistical entropy assigns the networks nearly identical complexity. The effect of the penalty is therefore not visible in global statistics, consistent with its acting locally on the 3×3 filters it targets.

L.5.3 Discussion

The two experiments show that a non-differentiable complexity penalty can be optimized directly within Hamming-SAM, improving test accuracy on top of sharpness-aware training. Characterizing how the penalty shapes the weights, for instance by tracking per-filter complexity during training, is a natural direction for future work.

M Additional Results

M.1 Tabular Version of Benchmark Results

Tables 7 and 8 show the full clean and corruption error rates for all methods and architectures on CIFAR-10 and CIFAR-100, averaged over 4 seeds. These complement the summary plots in § 7.

Table 7: CIFAR-10 clean and corruption error rates (%) across methods and architectures. Results averaged over 4 seeds (± std shown).

Model	Method	Clean Error (%) ↓	Corruption Error (%) ↓				
			Level 1	Level 2	Level 3	Level 4	Level 5
ResNet18	FP	5.38 ± 0.18	11.90 ± 0.10	17.53 ± 0.33	23.27 ± 0.22	30.87 ± 0.18	43.18 ± 0.19
	STE	7.94 ± 0.06	14.40 ± 0.08	20.38 ± 0.36	26.03 ± 0.75	33.57 ± 1.49	44.80 ± 1.58
	STE-SAM	7.76 ± 0.10	13.96 ± 0.13	20.35 ± 0.33	27.03 ± 0.66	35.68 ± 0.95	46.88 ± 1.03
	BOLD	6.99 ± 0.15	14.08 ± 0.12	20.69 ± 0.43	27.40 ± 0.56	35.44 ± 0.90	46.94 ± 0.64
	Hamming-SAM	**6.44** ± 0.17	**12.87** ± 0.10	**18.06** ± 0.18	**23.70** ± 0.34	**31.32** ± 0.50	**42.69** ± 0.42
VGG16	FP	7.07 ± 0.20	13.60 ± 0.16	19.47 ± 0.10	25.60 ± 0.08	33.18 ± 0.18	44.85 ± 0.44
	STE	8.72 ± 0.18	15.27 ± 0.28	21.40 ± 0.37	27.70 ± 0.44	35.77 ± 0.59	46.18 ± 0.67
	STE-SAM	8.81 ± 0.11	15.27 ± 0.17	21.31 ± 0.27	27.56 ± 0.44	35.63 ± 0.45	46.22 ± 0.52
	BOLD	8.25 ± 0.07	15.29 ± 0.15	21.57 ± 0.29	27.83 ± 0.33	35.82 ± 0.25	46.28 ± 0.19
	Hamming-SAM	**7.65** ± 0.08	**14.08** ± 0.10	**19.95** ± 0.10	**26.23** ± 0.26	**34.11** ± 0.30	**45.10** ± 0.45
ViT	FP	19.57 ± 0.14	24.60 ± 0.14	30.26 ± 0.14	34.62 ± 0.14	40.66 ± 0.14	47.74 ± 0.14
	STE	22.91 ± 0.59	27.51 ± 0.42	32.87 ± 0.21	36.95 ± 0.10	42.43 ± 0.32	48.34 ± 0.61
	STE-SAM	23.58 ± 0.56	28.12 ± 0.11	33.19 ± 0.43	37.18 ± 0.70	42.73 ± 0.94	48.72 ± 1.14
	BOLD	21.04 ± 0.14	24.84 ± 0.10	30.11 ± 0.23	34.15 ± 0.11	39.54 ± 0.09	46.11 ± 0.15
	Hamming-SAM	**20.29** ± 0.11	**24.04** ± 0.49	**29.24** ± 0.58	**33.03** ± 0.12	**38.37** ± 0.12	**44.71** ± 0.12

Table 8: CIFAR-100 clean and corruption error rates (%) across methods and architectures. Results averaged over 4 seeds (± std shown).

Model	Method	Clean Error (%) ↓	Corruption Error (%) ↓				
			Level 1	Level 2	Level 3	Level 4	Level 5
ResNet18	FP	24.67 ± 0.15	36.61 ± 0.34	45.37 ± 0.42	51.10 ± 0.41	57.91 ± 0.41	67.86 ± 0.27
	STE	30.27 ± 0.45	41.00 ± 0.38	49.82 ± 0.68	55.39 ± 0.69	62.13 ± 0.64	70.92 ± 0.60
	STE-SAM	30.36 ± 0.08	40.61 ± 0.04	49.35 ± 0.24	55.09 ± 0.30	61.79 ± 0.22	70.47 ± 0.12
	BOLD	28.29 ± 0.04	39.80 ± 0.07	48.62 ± 0.06	54.14 ± 0.07	60.46 ± 0.15	69.75 ± 0.02
	Hamming-SAM	**27.63** ± 0.03	**37.51** ± 0.05	**45.21** ± 0.03	**50.62** ± 0.02	**57.17** ± 0.07	**66.26** ± 0.39
VGG16	FP	29.10 ± 0.27	41.17 ± 0.42	49.60 ± 0.54	54.71 ± 0.51	60.80 ± 0.55	70.00 ± 0.45
	STE	35.35 ± 0.57	44.98 ± 0.37	53.16 ± 0.28	58.11 ± 0.20	64.04 ± 0.06	71.68 ± 0.13
	STE-SAM	35.77 ± 0.50	45.22 ± 0.15	53.13 ± 0.11	58.04 ± 0.21	63.73 ± 0.33	71.16 ± 0.43
	BOLD	32.35 ± 0.09	43.70 ± 0.07	52.37 ± 0.03	57.35 ± 0.10	63.30 ± 0.10	71.38 ± 0.10
	Hamming-SAM	**31.26** ± 0.11	**41.60** ± 0.18	**49.78** ± 0.18	**54.95** ± 0.20	**61.03** ± 0.26	**69.35** ± 0.24
ViT	FP	43.92 ± 0.01	50.66 ± 0.03	57.64 ± 0.14	61.46 ± 0.16	66.90 ± 0.12	72.57 ± 0.27
	STE	47.36 ± 0.23	53.77 ± 0.50	59.88 ± 0.57	63.40 ± 0.46	68.36 ± 0.39	73.19 ± 0.34
	STE-SAM	47.22 ± 0.18	53.25 ± 0.64	59.22 ± 0.81	62.81 ± 0.72	67.86 ± 0.62	72.71 ± 0.58
	BOLD	47.51 ± 0.11	53.10 ± 0.14	59.63 ± 0.05	63.18 ± 0.15	68.08 ± 0.19	72.92 ± 0.11
	Hamming-SAM	**45.37** ± 0.15	**50.49** ± 0.15	**56.66** ± 0.15	**60.39** ± 0.15	**65.59** ± 0.15	**71.02** ± 0.15

M.2 Hamming-SAM on an STE Backend

To test whether the gains depend on BOLD, we run Hamming-SAM on the STE backend used for the STE and STE-SAM baselines. The ascent flips the top-k bits ranked by $-w_i\,\partial\hat{\mathcal{L}}/\partial w_i$, by negating the corresponding latent weights. The descent is the same SGD step on the latent weights as in STE-SAM, with the gradient taken at the adversarial point. The two methods therefore differ only in the perturbation, and both use two backward passes per step. All other settings follow § P, and we use $k = 500$, the default of the BOLD experiments.

Table 9: CIFAR-10 results on the STE backend. Clean accuracy (%, mean ± std) and mean corruption accuracy on CIFAR-10-C (mCA, %). STE and STE-SAM are from Table 7 (4 seeds). Hamming-SAM uses $k = 500$ (4 seeds).

Model	Method	Clean Acc (%)	mCA (%)
ResNet-18	STE	92.06 ± 0.06	72.16
	STE-SAM	92.24 ± 0.10	71.22
	Hamming-SAM	**94.48** ± 0.18	**76.60**
VGG-16	STE	91.28 ± 0.18	70.74
	STE-SAM	91.19 ± 0.11	70.80
	Hamming-SAM	**92.80** ± 0.02	**76.26**

On the STE backend, Hamming-SAM improves clean accuracy over STE by 2.4 points on ResNet-18 and 1.5 points on VGG-16, while STE-SAM changes it by less than 0.2 points. Mean corruption accuracy improves by 4.4 and 5.5 points. The gains hold for every $k \in \{50, 100, 500, 1000\}$. Across this range, clean accuracy stays between 94.21 and 94.77 on ResNet-18 and between 92.19 and 93.31 on VGG-16, and mCA stays between 75.31 and 76.82 and between 75.68 and 76.26. The discrete perturbation therefore improves both accuracy and robustness without BOLD, while latent-space SAM barely changes the STE baseline. Fig. 9 shows that the gap to STE and STE-SAM is larger at strong corruption levels than on clean data. Fig. 10 shows the effect of k. Clean accuracy is highest at $k = 100$ and decreases slightly for larger k, while mCA is highest at $k = 500$ or 1000. This matches the bias–sharpness tradeoff observed on the BOLD backend (§ 7.2).

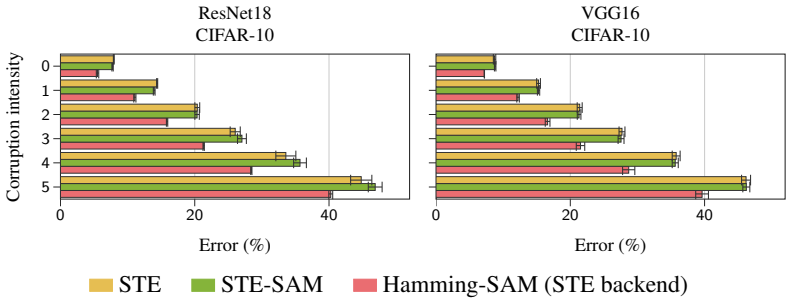


Figure 9: Clean (level 0) and CIFAR-10-C error at each corruption intensity on the STE backend. Hamming-SAM uses $k = 500$. Error bars show the standard deviation over seeds.

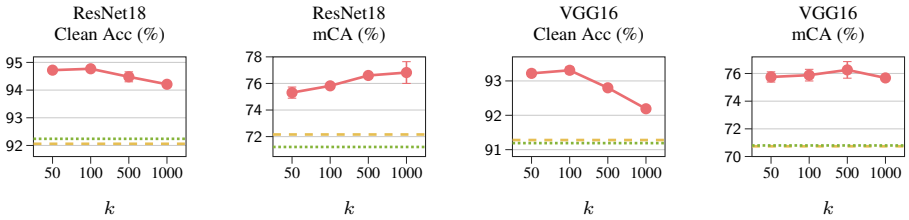
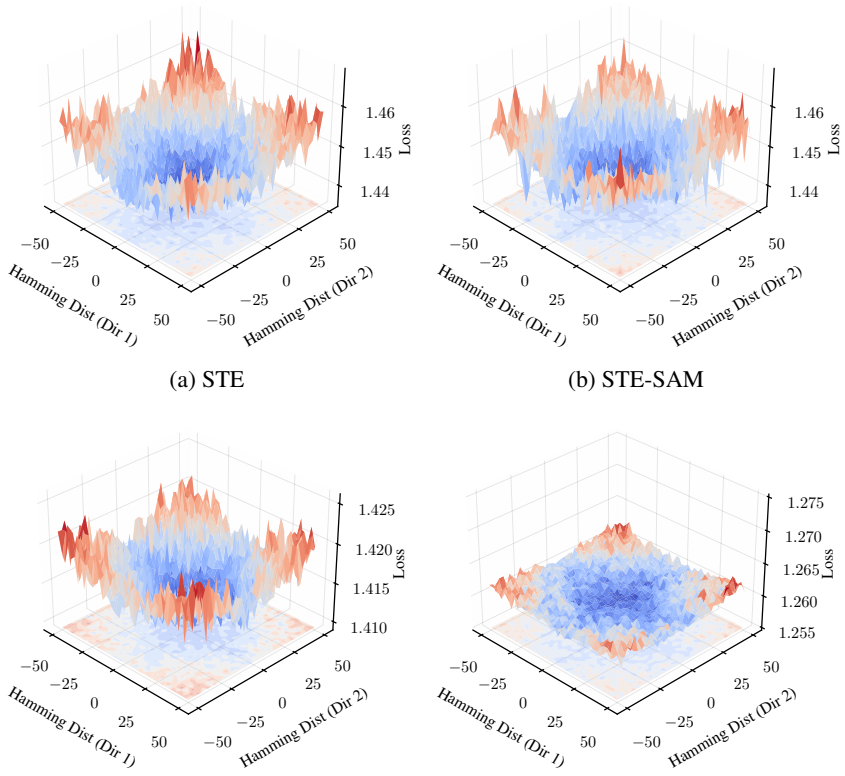


Figure 10: Effect of the Hamming radius k on the STE backend. Red: Hamming-SAM (mean ± std over 4 seeds). Dashed: STE. Dotted: STE-SAM.

M.3 Visualizing the Loss Landscape



(a) STE (b) STE-SAM

(c) BOLD (d) Hamming-SAM

Figure 11: **Empirical 3D loss landscapes of binary and Boolean MLPs perturbed along two random discrete Hamming directions, evaluated on a test set. (a, b, c)** Standard methods, including STE, STE-SAM, and standard BOLD, stay in high-loss local minima with rugged surfaces. **(d)** Hamming-SAM changes the optimization trajectory. By reducing discrete curvature, it leaves the rugged plateau and finds a deeper and wider basin that generalizes better.

Methodology for 2D Hamming subspace projection. To visualize the discrete optimization landscape of the MLP (3 hidden layers, 128 units), we project the high-dimensional loss surface onto a 2D Hamming subspace. Moving in a discrete parameter space requires flipping discrete bits, that is, changing the sign of the binary or Boolean weights.

We construct two random, orthogonal perturbation directions. For the discrete weights of the network, we generate two master masks with 50% density, which are subsequently orthogonalized to ensure zero overlap. Each mask is split into positive and negative sets of indices to allow for bidirectional perturbation. Let $h_1, h_2 \in [-k_{\max}, k_{\max}]$ represent the discrete Hamming distance (number of bit flips) taken along these two orthogonal directions. For our high-resolution visualizations, we use $k_{\max} = 50$ and scan a 41×41 grid of evenly spaced flip counts.

To average out the noise of random bit-flipping, we apply Monte Carlo smoothing. At each grid point (h_1, h_2), we draw $N = 10$ independent random permutations of the target bits, flip the corresponding weights $w \leftarrow -w$, evaluate the test loss, and record the expected (mean) loss $\mathbb{E}[L(h_1, h_2)]$.

Quantitative flatness metric. To quantitatively assess the topology of the visualized basins, we define a scalar flatness score as the Mean Absolute Error (MAE) between the expected perturbed losses across the grid and the base unperturbed loss $L(0, 0)$:

$$\text{Flatness} = \frac{1}{|G|} \sum_{(h_1,h_2)\in G} |\mathbb{E}[L(h_1, h_2)] - L(0, 0)| \tag{83}$$

where G is the set of all 1,681 evaluated discrete grid points. A lower score indicates a flatter, more stable basin.

Landscape analysis. As illustrated in Fig. 11, standard methods, including STE, STE-SAM ($\rho = 0.05$), and standard BOLD, stay on non-convex, rugged plateaus at a high test loss level. The flatness score confirms this. STE achieves a flatness score of 0.0044, STE-SAM scores 0.0045, and BOLD slightly improves to 0.0039. Note that latent-space SAM does not flatten the discrete landscape at all here, scoring marginally worse than plain STE, which is the behavior the vanishing perturbation effect of § 3 predicts.

In contrast, Hamming-SAM ($k = 500$) reduces discrete curvature, which lets it leave this rugged plateau and reach a deeper, flatter minimum. The flatness score for Hamming-SAM drops to 0.0011, roughly a $3.5\times$ reduction relative to standard BOLD and a $4\times$ reduction relative to STE. This visualization shows that Hamming-SAM changes the discrete geometry and finds a basin that supports both optimization and robust generalization.

N Empirical Verification of Theoretical Predictions

We verify the key assumptions and predictions of the theoretical framework (Theorems 5.2, 5.7 and 5.8 and Proposition 5.11) by measuring discrete landscape quantities at training checkpoints across three architectures: a 3-layer MLP ($n = 442{,}368$), ResNet-18 ($n = 11{,}157{,}504$), and VGG-16 ($n = 14{,}708{,}736$), all trained on CIFAR-10 with standard BOLD and Hamming-SAM. All quantities are computed on the training data for consistency, and use exact finite differences rather than the first-order estimate of § 4.2.

N.1 Submodularity Ratio γ_k (Theorem 5.2)

Table 10 reports the submodularity ratio γ_k, estimated by sampling 100 random (S, L) pairs per checkpoint (with $|S| \leq k$ and $|L| \leq 3$). A visual illustration of γ_k evolution is shown in Fig. 12.

The median γ_k is 1.000 at every checkpoint but one (ResNet-18 with Hamming-SAM at epoch 150, where it is 0.998), meaning the loss landscape is exactly submodular in the majority of sampled configurations and the greedy ascent achieves the full $(1-1/e)$ approximation ratio there. The minimum γ_k never falls below 0.50, and at convergence it is at least 0.84 on the MLP and at least 0.98 on the convolutional architectures. The lowest value ($\gamma_{\min} = 0.50$) occurs for standard BOLD on the MLP at epoch 1 (random initialization). By epoch 300, $\gamma_{\min}$ rises to 0.92, indicating the landscape becomes more submodular during training. The same upward trend holds on the convolutional architectures (e.g., VGG with Hamming-SAM: $0.92 \rightarrow 0.99$), consistent with the landscape becoming more submodular as the optimizer converges to a flat basin. What the guarantee of Theorem 5.2 requires is only that γ_k be bounded away from zero, which holds with a wide margin throughout.

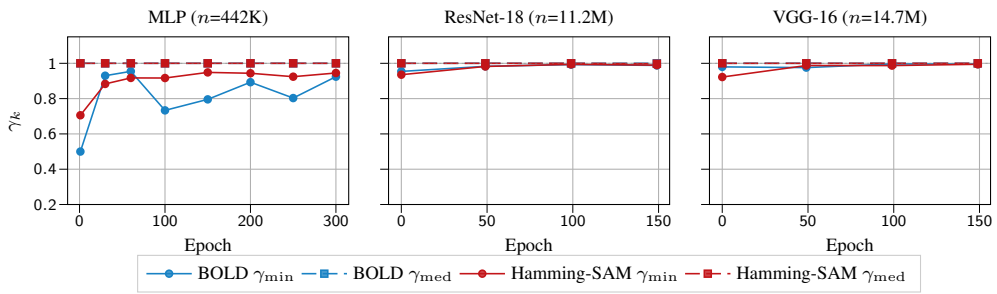


Figure 12: Submodularity ratio γ_k over training. Solid lines: $\gamma_{\min}$. Dashed lines: γ_{med}.

N.2 Discrete Hessian Spectrum (Theorem 5.8)

Table 11 reports the evolution of the discrete Hessian H_{ij} during training, estimated by sampling 300 random (i, j) pairs per checkpoint. Each entry is an exact finite difference of the cross-entropy on a

Table 10: Submodularity ratio γ_k at initialization and convergence. Estimated from 100 random (S, L) samples per checkpoint.

Architecture	Method	Epoch	$\gamma_{\min}$	γ_{med}	γ_{mean}
MLP (n=442K)	BOLD	1	0.500	1.000	1.001
	BOLD	300	0.924	1.000	1.000
	Hamming-SAM	1	0.906	1.000	1.002
	Hamming-SAM	300	0.845	1.000	0.998
ResNet-18 (n=11.2M)	BOLD	0	0.954	1.000	1.004
	BOLD	150	0.988	1.000	1.002
	Hamming-SAM	0	0.935	1.000	1.002
	Hamming-SAM	150	0.990	0.998	0.997
VGG-16 (n=14.7M)	BOLD	0	0.980	1.000	1.002
	BOLD	150	0.995	1.000	1.000
	Hamming-SAM	0	0.922	1.000	0.997
	Hamming-SAM	150	0.995	1.000	1.001

fixed subset of 1,024 training images, computed in single precision, so entries below about 10^{-8} are indistinguishable from zero.

Interaction scale. At convergence, kh ranges from 1.3×10^{-4} (ResNet-18 with Hamming-SAM) to 2.3×10^{-2} (MLP with BOLD), which we use as a measure of the sampled interaction scale. The condition $\eta > kh$ of Theorem 5.8 refers to a fixed threshold on the exact Boolean Variation, whereas the accumulator thresholds the accumulated backward signal with an adaptive threshold (§ I), so the two are not directly comparable. Theorem 5.8 involves only the pairs formed by the descent bit and the ascent set. § N.4 shows that interactions among the selected coordinates are of the same order as among random pairs, so the random-pair statistic is a reasonable proxy for the one entering the theorem.

Hamming-SAM converges to low-h regions. On ResNet-18, $h_{\max}$ drops by about two orders of magnitude, from 1.2×10^{-6} (epoch 0) to 1.2×10^{-8} (epoch 150) under Hamming-SAM, close to the resolution of the loss, with the fraction of entries below 10^{-8} rising from 11% to 99%. VGG-16 shows a similar pattern: $h_{\max}$ drops by 12× from 1.7×10^{-6} to 1.4×10^{-7}, with this fraction rising from 19% to 93%. On both architectures, the fraction for same-layer pairs reaches 88–100% at convergence, confirming the structural attenuation discussed in Remark 5.10.

Standard BOLD also shows decreasing $h_{\max}$, but the reduction is smaller and $h_{\max}$ remains an order of magnitude larger than that of Hamming-SAM at convergence on ResNet-18.

Table 11: Discrete Hessian evolution during training. $h_{\max} = \max |H_{ij}|$ over 300 sampled pairs, $\%_0$ = fraction with $|H_{ij}| \leq 10^{-8}$, and $kh = k \cdot h_{\max}$.

Arch.	Method	Epoch	$h_{\max}$	$\bar{h}$	$\%_0$	kh
MLP	BOLD	1	1.8×10^{-6}	2.5×10^{-7}	8%	7.9×10^{-4}
	BOLD	300	5.1×10^{-5}	2.3×10^{-6}	3%	2.3×10^{-2}
	Hamming-SAM	1	7.2×10^{-7}	1.6×10^{-7}	15%	3.2×10^{-4}
	Hamming-SAM	300	3.1×10^{-6}	6.2×10^{-7}	7%	1.4×10^{-3}
ResNet-18	BOLD	0	5.1×10^{-7}	1.0×10^{-7}	11%	5.6×10^{-3}
	BOLD	150	2.5×10^{-7}	1.1×10^{-8}	76%	2.8×10^{-3}
	Hamming-SAM	0	1.2×10^{-6}	9.8×10^{-8}	11%	1.4×10^{-2}
	Hamming-SAM	150	1.2×10^{-8}	1.5×10^{-9}	99%	1.3×10^{-4}
VGG-16	BOLD	0	1.2×10^{-6}	7.6×10^{-8}	16%	1.8×10^{-2}
	BOLD	150	3.4×10^{-7}	5.5×10^{-9}	93%	5.0×10^{-3}
	Hamming-SAM	0	1.7×10^{-6}	7.5×10^{-8}	19%	2.4×10^{-2}
	Hamming-SAM	150	1.4×10^{-7}	3.3×10^{-9}	93%	2.1×10^{-3}

N.3 Multi-Bit Descent (Proposition 5.11)

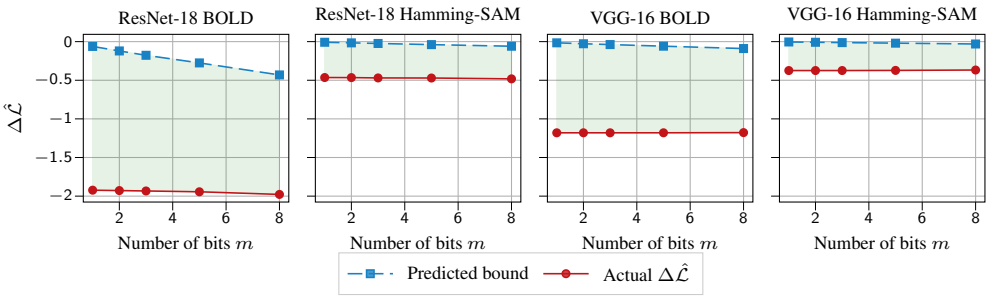


Figure 13: Multi-bit descent verification at convergence. Dashed blue: predicted upper bound $-m\delta + \binom{m}{2}h$. Solid red: actual loss change. The bound holds whenever actual $\leq$ predicted (ResNet-18 and VGG-16: 40/40).

Proposition 5.11 predicts that simultaneously flipping m bits, each with $\mathrm{BV}_{i_l}(w^t) < -\delta$, decreases the loss provided $\delta > (m-1)h/2$. Fig. 13 reports the predicted bound $-m\delta + \binom{m}{2}h$ versus the actual loss change for $m \in \{1, 2, 3, 5, 8\}$ at the final checkpoint.

On ResNet-18 and VGG-16, the bound holds for all m values at all checkpoints: 40/40 tests across both architectures. The actual loss decrease is consistently much larger than the predicted bound (e.g., for VGG with Hamming-SAM at epoch 150, predicted ≤ -0.031 versus actual $= -0.368$ for $m = 8$). The bound is therefore conservative but valid.

N.4 Sequential vs. Parallel Greedy Ascent Quality

Table 12: Adversarial loss increase: top-k versus sequential greedy (Hamming-SAM only, $k = \lfloor 10^{-3}n \rfloor$). Ratio is $f_{\text{seq}}/f_{\text{top-k}}$, $h_{\text{eff}} = (f_{\text{seq}} - f_{\text{top-k}})/(k(k-1))$ is the pairwise interaction that would account for the whole deficit (–: no deficit, as top-k exceeds the sequential greedy), and $h_{\max}$ is the random-pair estimate of Table 11.

Architecture	Epoch	$f_{\text{top-k}}$	f_{seq}	Ratio	h_{eff}	$h_{\max}$
MLP (k=442)	1	0.12	0.36	2.8×	1.2×10^{-6}	7.2×10^{-7}
	300	0.20	0.74	3.7×	2.8×10^{-6}	3.1×10^{-6}
ResNet-18 (k=11,157)	0	5.38	6.20	1.2×	6.6×10^{-9}	1.2×10^{-6}
	150	6.35	8.90	1.4×	2.0×10^{-8}	1.2×10^{-8}
VGG-16 (k=14,708)	0	12.78	8.31	0.7×	–	1.7×10^{-6}
	150	2.37	5.21	2.2×	1.3×10^{-8}	1.4×10^{-7}

Theorem 5.2 analyzes the sequential greedy, while Algorithm 1 uses a top-k variant for efficiency, and Theorem 5.4 bounds the resulting deficit. Table 12 compares the adversarial loss increase achieved by both. The sequential greedy finds a stronger adversarial point at every checkpoint except VGG-16 at initialization, where the one-shot selection happens to pick a better set, which is possible because neither rule is optimal. This is consistent with Theorem 5.4, which attributes the deficit of the one-shot rule to interactions among the selected coordinates. The top-k variant still identifies meaningful adversarial configurations in all cases ($f_{\text{top-k}} > 0$ everywhere), retaining 71% of the sequential gain on ResNet-18 and 45% on VGG-16 at convergence. The deficit is smallest on ResNet-18 (1.2–1.4×) and largest on the MLP (2.8–3.7×).

Relation to the top-k bound. Theorem 5.4 bounds the deficit by the interactions among the selected coordinates. These may differ from the random-pair statistics of Table 11, since the ascent selects the k largest of n Boolean Variations. Table 12 compares the two through the effective interaction h_{eff}. This is the pairwise interaction that would account for the whole deficit. In every case, h_{eff} is below the sampled $h_{\max}$ or within a factor of two of it. Interactions among the selected coordinates are therefore of the same order as among random pairs. At convergence, the bound instantiated

with the sampled $h_{\max}$ covers the measured deficit on the MLP (0.60 against 0.54) and on VGG-16 (30 against 2.8). On ResNet-18, it accounts for most of the deficit (1.4 against 2.55). Theorem 5.4 attributes the remainder to interactions of order three and higher.

O Limitations and Future Work

Our evaluation covers standard benchmarks used in the SAM literature (CIFAR-10/100, ImageNet) across diverse architectures (MLP, VGG, ResNet, ViT), complemented by extensive theoretical verification of the discrete landscape quantities that underpin the framework. We record below the assumptions that the analysis leaves open, followed by two natural extensions.

Beyond binary weights. The theoretical framework, comprising Hamming-ball perturbations, the discrete Hessian and the submodularity ratio, extends naturally to ternary $\{-1, 0, +1\}$ or multi-bit quantized weights, where the parameter space remains a discrete lattice. Developing Hamming-SAM variants for these settings could broaden the practical impact to mixed-precision and low-bit quantization regimes.

Large-scale language models. Recent work has demonstrated the feasibility of Boolean and low-bit large language models [55, 54]. Evaluating Hamming-SAM at this scale would test whether the sharpness-robustness connection observed in our experiments extends to billion-parameter architectures. This is beyond our current computational resources but is a promising direction.

P Experimental Details

Architectures. We follow standard practices in the BNN and BOLD literature [26, 42]. For all binary methods, full-precision linear and convolutional layers are replaced with their Boolean (BOLD) or binary (STE) counterparts without modifying the network architecture or introducing additional learnable parameters. We evaluate ResNet-18 and ResNet-50 [22], VGG-16 [52], ViT-Tiny [15], and a 3-layer MLP with 128 hidden units per layer.

Training protocol. All experiments were implemented in Python using PyTorch [46]. For CIFAR-10/100 [31], models were trained for 150 epochs with a minibatch size of 128. For ImageNet [13], models were trained for 90 epochs with a minibatch size of 512. All runs use a cosine learning rate scheduler.

Optimizer settings. For STE-based methods (STE, STE-SAM), we use SGD with a single momentum term for fair comparison with the BOLD accumulator, which also uses single momentum. The learning rate was selected from $\{0.01, 0.05, 0.1\}$. We also tried Adam and found similar results. BOLD and Hamming-SAM use the BOLD accumulator [42] with the learning rate selected from $\{10, 30, 50\}$.

Hyperparameter selection. For Hamming-SAM, the Hamming radius k was selected from $\{50, 100, 500, 1000\}$ on a held-out validation set. For STE-SAM, the perturbation radius ρ was selected from $\{0.005, 0.01, 0.05, 0.1, 0.5\}$ using the same protocol. All other hyperparameters (learning rate, batch size, number of epochs) were kept identical across methods for fair comparison.

Evaluation. Clean accuracy is reported on the standard test split. Corruption robustness is evaluated on CIFAR-10-C/100-C [24] (19 corruption types $\times$ 5 severity levels) and ImageNet-C [24]. We report mean corruption accuracy (mCA) averaged over all types and severities. OOD detection on ImageNet uses AUROC. All results are averaged over 4 seeds with standard deviations reported.

Computational cost. Each Hamming-SAM step uses two backward passes, one at w^t and one at $\hat{w}$, as in continuous SAM. On a single V100, a Hamming-SAM step takes about $1.9\times$ the time of a standard BOLD step, both for VGG-16 on CIFAR-10 and for ResNet-50 on ImageNet. The top-k selection uses PyTorch's `topk` and adds less than 1% to the step time. The only extra memory is the list of at most k flipped indices, which takes a few kilobytes.

Q Illustrative Example

Algorithm 2 provides an illustraive example of the Hamming-SAM optimizer in Python using PyTorch [46]. It reads a per-bit signal from `p.grad`, so any backend can be used. For STE (§ M.2), pass the latent weights and a standard optimizer as `descent`.

Algorithm 2: Illustrative Python code of Hamming-SAM optimizer

```python
import torch


class HammingSAM(torch.optim.Optimizer):
    """Hamming-SAM on {-1, +1}^n. p.grad holds any per-bit signal f'_i (BOLD
    backward pass or STE gradient), so that BV_i ~ -sign(w_i) * f'_i.

    Args:
        params: Boolean weights in {-1, +1} (BOLD) or latent weights (STE).
        lr: Learning rate of the default descent step (BOLD accumulator).
        k: Hamming radius.
        descent: Callable descent(group, i), or a torch optimizer on the latent
            weights (STE), e.g. descent=torch.optim.SGD(w, lr=0.05).
    """
    def __init__(self, params, lr, k=500, descent=None):
        super().__init__(params, dict(lr=lr))
        self.k = k
        self.descent = descent or self._accumulator_update
        for g in self.param_groups:
            g['accums'] = [torch.zeros_like(p) for p in g['params']]
            g['ratios'] = [1.0 for _ in g['params']]
            g['adv_flips'] = []

    @torch.no_grad()
    def first_step(self, zero_grad=False):
        """Phase 1: greedy ascent, flip the top-k bits with positive BV."""
        bv = torch.cat([(-p.data.sign() * p.grad.data).view(-1)
                        for g in self.param_groups
                        for p in g['params'] if p.grad is not None])

        vals, topk = torch.topk(bv, min(self.k, bv.numel()))
        topk = set(topk[vals > 0].cpu().tolist())  # only BV_i > 0

        offset = 0
        for g in self.param_groups:
            g['adv_flips'] = []
            for i, p in enumerate(g['params']):
                if p.grad is None: continue
                idx = [j - offset for j in topk
                       if offset <= j < offset + p.numel()]
                if idx:
                    idx = torch.tensor(idx, device=p.device)
                    g['adv_flips'].append((i, idx))
                    p.view(-1)[idx] *= -1
                offset += p.numel()

        if zero_grad: self.zero_grad()

    @torch.no_grad()
    def second_step(self, zero_grad=False):
        """Phase 2: restore, then descend with the signal at w_adv."""
        for g in self.param_groups:  # restore the flipped bits
            for i, idx in g.get('adv_flips', []):
                g['params'][i].view(-1)[idx] *= -1
        if isinstance(self.descent, torch.optim.Optimizer):
            self.descent.step()  # STE backend: latent update at w_adv
        else:
            for g in self.param_groups:
                for i, p in enumerate(g['params']):
                    if p.grad is not None:
                        self.descent(g, i)

        if zero_grad: self.zero_grad()

    @staticmethod
    def _accumulator_update(g, i):
        """Example Descent with BOLD accumulator step (Algorithm 8 in BOLD paper)."""
        p, a = g['params'][i], g['accums'][i]
        a.mul_(g['ratios'][i]).add_(p.grad, alpha=g['lr'])
        flip = a * p >= 1
        p[flip] *= -1
        a[flip] = 0
        g['ratios'][i] = 1.0 - flip.float().mean().item()
```

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions claimed in the abstract and introduction are fully based on the method described in § 4, § 5, and empirical results in § 7.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations and future work in § O.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumption of each theoretical result is explicitly and formally stated. However, due to space constraint, all the proofs are postponed to the Appendix.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed algorithm is described in § 4. We also provide detailed experimental settings and hyperparameters in § P.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We used publicly common datasets and standard model architectures in our experiments. In the Appendix, we provide example code and experimental details for reproduction purposes.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: All details can be found in § P.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We repeat experiments with different random seeds and report the mean and standard deviation of the results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We mention it in the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: It has been and continues to be our first priority to be fully compliant with the scientific research spirit, and with NeurIPS Code of Ethics in particular.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.

- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We used publicly common datasets and code package. We fully cited the original paper that produced those assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: NA.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: NA.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: NA.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: NA.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.