GFNAS: Towards Gradient-Friendly Neural Network Architecture Search From Sharpness-Aware Perspective

1st Jianlun Ma
College of software,
Northeastern University, China
2490182@stu.neu.edu.cn

2nd Xinxin Xu
College of software,
Northeastern University, China
xuxx1@mails.neu.edu.cn

3rd Lianbo Ma
College of software,
Northeastern University, China
Foshan Graduate School of Innovation,
Northeastern University, Foshan, China
malb@swc.neu.edu.cn

***Abstract*—Neural architecture search (NAS) aims to automatically discover high-performing network architectures for specific tasks. In recent years, NAS methods that leverage sharpness signals have shown promising potential for improving model generalization. However, NAS still falls short in robustness under out-of-distribution (OOD) shifts. Recent studies have incorporated Sharpness-Aware Minimization (SAM) into the search process, but under weight-sharing and short training-budget settings, the standard SAM perturbation direction may entangle deterministic gradient trends with mini-batch stochastic noise, thereby biasing the sharpness proxy. To address this issue, we propose GFNAS, a gradient-friendly sharpness-aware NAS framework that estimates the deterministic gradient trend via an exponential moving average (EMA) and suppresses it when constructing perturbations, yielding a noise-consistent direction. Without increasing the search cost or modifying the search space, this strategy provides a more robust robustness-oriented evaluation signal. Experiments on CIFAR-10/100 and their corrupted variants demonstrate that GFNAS improves both accuracy and corruption robustness (mCE), reduces search cost, and alleviates sensitivity to sharpness hyperparameters.**

***Index Terms*—Neural Architecture Search (NAS), Sharpness-Aware Minimization (SAM), Stochastic gradient noise**

I. Introduction

Neural Architecture Search (NAS) [27], [31] has become an effective paradigm for automating network design, enabling the discovery of architectures that perform well under diverse computational and deployment constraints. However, despite strong results on in-distribution (ID) benchmarks, conventional NAS methods often exhibit poor robustness and limited generalization under distribution shifts [5], [38], such as common corruptions or unseen environments. This reveals a fundamental limitation of NAS: architectures optimized for high ID accuracy do not necessarily achieve robust or stable generalization [6], [36].

Recent studies suggest that the geometry of the loss landscape plays a key role in generalization [16], [30]. In particular, models converging to flatter minima often show better robustness and out-of-distribution (OOD) performance [11], [12], [35]. Based on this insight, sharpness-aware optimization methods, especially Sharpness-Aware Minimization (SAM) [7], explicitly steer training toward flatter regions of the loss surface. By minimizing the worst-case loss within a local parameter neighborhood, SAM has consistently improved both generalization and robustness across a range of tasks.

Building on this insight, FlatNAS [9] incorporates sharpness-aware training directly into NAS. Rather than using optimization only as a post-search refinement, it adopts SAM-based training as the evaluator during search and introduces a robustness-aware objective that combines classification accuracy with sensitivity to parameter perturbations. This allows FlatNAS to identify architectures that are inherently more robust to weight perturbations, yielding better OOD robustness than conventional accuracy-driven NAS methods. More importantly, it shows that the optimizer is not merely an implementation detail, but a key factor shaping the search trajectory [10], [37].

Despite its effectiveness, FlatNAS implicitly assumes that the sharpness signal induced by SAM is a faithful and stable proxy for architecture robustness. Recent advances in sharpness-aware optimization question this assumption [1]. FriendlySAM [19] shows that SAM's perturbation direction mixes gradient components with different effects on generalization: the full-batch gradient tends to dominate the perturbation direction, whereas stochastic gradient noise from mini-batch sampling plays a more important role in improving generalization [2], [29]. By decoupling these components and suppressing deterministic bias with an exponential moving average (EMA), FriendlySAM achieves more stable sharpness estimation, better robustness, and lower sensitivity to the perturbation radius.

This observation exposes a previously overlooked limitation in sharpness-aware NAS frameworks: not all sources of sharpness contribute equally to robust architecture evaluation. In the context of NAS, where architectures are compared based on short-horizon training signals, deterministic gradient components can introduce systematic bias, potentially distorting the relative ranking of candidate architectures [4].

Corresponding author: Lianbo Ma. This work is supported by National Natural Science Foundation of China under Grant 62472079.

As a result, architectures favored under standard SAM-based evaluation may not correspond to those with genuinely robust loss landscapes.

In this work, we revisit sharpness-aware neural architecture search from the perspective of gradient component decomposition. We propose a noise-consistent sharpness-aware NAS framework that refines the search evaluator by isolating stochastic sharpness signals and suppressing deterministic gradient bias. Concretely, we reformulate the inner-loop optimization of FlatNAS with a noise-consistent sharpness-aware training strategy, while preserving the original search space and computational budget. This yields a more faithful robustness proxy for architecture evaluation and enables the discovery of architectures with consistently better OOD performance.

Extensive experiments show that our method not only improves robustness metrics, including corruption error and sensitivity to parameter perturbations, but also enhances search stability across different hyperparameter settings. In particular, it substantially reduces sensitivity to the perturbation radius and produces Pareto fronts that dominate those of prior sharpness-aware NAS methods. These results indicate that noise-consistent sharpness estimation is a critical yet previously underexplored factor in robust NAS. Our main contributions are summarized as follows:

- We thoroughly analyzed the impact of full gradients and stochastic gradients on the NAS process from the perspective of adversarial perturbation composition.
- We propose a Gradient Friendly Neural Network Architecture Search via Sharpness-Awareness (GFNAS), enhancing the generalization of architectures.
- Extensive experimental results indicate that the proposed GFNAS method effectively improves generalization capabilities and facilitates faster convergence of the algorithm.

II. Related Work

A. Neural Network Architecture Robustness

In the context of neural architecture search for adversarial robustness, several approaches build upon the widely adopted DARTS framework [21], including RNAS [39], NADAR [20], and RACL [5]. RNAS aims to improve adversarial robustness by introducing a regularization strategy that enforces consistency between the network outputs obtained from clean samples and their adversarial counterparts, encouraging architectures with more stable predictions under worst-case perturbations. In a related manner, NADAR focuses on enhancing robustness of neural networks under floating-point operation (FLOPs) constraints. To achieve this, NADAR augments backbone architectures with additional robustness-oriented layers while explicitly accounting for the computational overhead introduced by such architectural expansions, thereby balancing robustness gains against efficiency costs [21].

B. Sharpness-Aware Optimization

The geometry of the loss landscape is a key factor in generalization, with flatter minima often linked to better robustness [3], [13]. Sharpness-Aware Minimization (SAM) follows this principle by optimizing the worst-case loss within a local parameter neighborhood to encourage flatter solutions [8], [23]. FriendlySAM [19] further shows that SAM's perturbation direction mixes deterministic full-gradient components with stochastic gradient noise, which affect generalization differently. By suppressing deterministic bias, FriendlySAM achieves more consistent sharpness estimation and improved robustness in standard fixed-architecture training. However, existing sharpness-aware optimizers are mainly studied in conventional training settings, rather than as evaluation signals beyond final model optimization [15].

C. Sharpness-Aware Neural Architecture Search

Inspired by the connection between sharpness and generalization, recent work [9], [14], [22] has begun to integrate sharpness-aware optimization into neural architecture search. FlatNAS [9] represents a notable advance in this direction by embedding SAM-based training directly into the architecture search process. Rather than treating sharpness-aware optimization as a post-search refinement step, FlatNAS employs SAM as the evaluator during search and introduces a robustness-aware objective that jointly considers classification accuracy and sensitivity to parameter perturbations. While this design enables the discovery of architectures that are intrinsically robust to weight perturbations, it implicitly assumes that the sharpness signal induced by standard SAM provides a reliable proxy for comparing candidate architectures. In practice, NAS relies on short-horizon training under limited computational budgets, where optimization-induced bias may dominate evaluation signals [25], [32]. Consequently, the fidelity of SAM-based sharpness estimates for architecture ranking remains an open question, particularly when different sources of sharpness contribute unequally to generalization.

III. Method

A. Problem Formulation

We consider the problem of neural architecture search under a differentiable optimization framework. Let D_{tra} and D_{val} denote the training and validation datasets, respectively. The architecture search space Ω is defined following the standard cell-based differentiable NAS paradigm. Specifically, an architecture $x \in \Omega$ is represented as a directed acyclic graph (DAG), where each edge corresponds to a mixed operation selected from a predefined candidate set:

$$\mathcal{O} = \{o_1, o_2, \ldots, o_K\} \tag{1}$$

including common operators such as separable convolutions, dilated convolutions, pooling, and identity mappings. To enable gradient-based optimization, the discrete architecture is relaxed into a continuous representation parameterized by architecture variables α. For each edge (i, j) in the DAG, the mixed operation is expressed as a softmax-weighted combination:

$$\bar{o}_{(i,j)}(x) = \sum_{k=1}^{K} \frac{\exp\left(\alpha_{(i,j)}^{k}\right)}{\sum_{k'} \exp\left(\alpha_{(i,j)}^{k'}\right)} o_k(x), \tag{2}$$

where k denotes the index of candidate operations in the search space. Under this relaxation, the network output can be written as:

$$f(\cdot, \omega, \alpha) \tag{3}$$

where ω denotes the shared network weights.

The architecture search problem is formulated as a bi-level optimization:

$$\begin{aligned} &\min_{\alpha} \; L_{\text{val}}\left(\omega^{\star}(\alpha), \alpha\right) + \lambda \phi(C(\alpha) - \tau), \\ &\text{s.t. } \omega^{\star}(\alpha) = \arg\min_{\omega} L_{tra}(\omega, \alpha), \end{aligned} \tag{4}$$

where L_{tra} and L_{val} denote the empirical risk evaluated on the training and validation sets, respectively, $\phi(\cdot)$ is a penalty function enforcing the resource constraint, λ balances accuracy and efficiency. The function $C(\alpha)$ measures the expected computational complexity of the architecture induced by α, and τ specifies a target resource budget. In practice, $C(\alpha)$ can be instantiated as a differentiable proxy of resource usage, such as expected FLOPs or parameter count:

$$C(\alpha) = \sum_{(i,j)} \sum_{k=1}^{K} \pi_{(i,j)}^{k}(\alpha) \cdot c\left(o_k\right), \tag{5}$$

where $\pi_{(i,j)}^{k}(\alpha)$ denotes the softmax probability of selecting operation o_k on edge (i, j), and $c(o_k)$ is the predefined cost (e.g., FLOPs) associated with operation o_k. This formulation decouples architecture selection from parameter optimization, while enabling gradient-based optimization over α.

B. Sharpness-Aware Neural Architecture Search

Standard differentiable NAS evaluates candidate architectures by training shared weights for a limited number of steps and comparing validation performance. Under weight-sharing and short-horizon training, however, the learned weights are often highly sensitive to parameter perturbations, indicating sharp loss regions. This leads to unstable performance estimation and unreliable architecture ranking.

To improve evaluation reliability, we introduce sharpness-aware optimization into the inner loop. Instead of minimizing training loss at a single parameter point, we seek weight solutions that remain stable within a local neighborhood. For an architecture parameterized by α, Eq. 6 is replaced by

$$\omega^{\star}(\alpha) = \arg\min_{\omega} \max_{|\epsilon| 2 \leq \rho} Ltra(\omega + \epsilon, \alpha), \tag{6}$$

where $\rho{>}0$ is the neighborhood radius and ϵ is a weight perturbation.

This objective discourages solutions with large loss variations under small perturbations, promoting flatter minima that are empirically linked to better generalization and robustness. In NAS, since α is updated according to the validation performance of $\omega(\alpha)$, sharpness-aware inner-loop optimization reduces sensitivity to initialization, stochastic noise, and limited training budgets, making $L_{val}(\omega(\alpha), \alpha)$ a more reliable proxy for architecture comparison. Integrating Eq. 6 into the bi-level optimization in Section III-A therefore yields a sharpness-aware NAS framework based on locally stable weight solutions.

C. Gradient Friendly NAS via SAM

1) Motivation: Although SAM improves the stability of architecture evaluation as discussed in III-B, directly adopting standard SAM-based training in NAS remains suboptimal. In practice, the perturbation direction used in SAM is computed from the gradient of a single mini-batch, which implicitly mixes multiple gradient components with distinct statistical properties.

In the NAS scenario, where architectures are evaluated under shared weights and limited training steps, such mixed perturbation directions can lead to biased sharpness estimates and unstable architecture ranking [33]. Specifically, perturbations dominated by deterministic gradient components tend to introduce systematic bias across architectures, masking the stochastic variations that are critical for assessing robustness under short-horizon training.

This phenomenon suggests that not all sources of sharpness contribute equally to reliable architecture evaluation, motivating a refined treatment of perturbation construction in sharpness-aware NAS.

2) Principle: Let $\nabla_{\omega} L_{tra}(\omega, \alpha)$ denote the stochastic gradient computed on a mini-batch sampled from D_{tra}. This gradient can be conceptually decomposed into two components:

- **a deterministic component aligned with the full-data gradient;**
- **a stochastic component induced by mini-batch sampling.**

Recent optimization studies indicate that the stochastic component plays a dominant role in promoting generalization, while the deterministic component may even impair robustness by amplifying sharpness at the dataset level [34]. In the context of NAS, this imbalance is particularly harmful, as architecture comparison relies on relative performance differences observed under noisy and resource-constrained training dynamics.

Therefore, an effective sharpness-aware evaluator for NAS should emphasize stochastic sharpness signals while suppressing deterministic bias, yielding perturbations that better reflect architecture-level robustness rather than optimization artifacts.

3) Gradient-Friendly Perturbation Construction: Based on the above insight, we propose a gradient-friendly sharpness-aware optimization strategy tailored for NAS. Instead of directly using the raw mini-batch gradient to construct adversarial perturbations, we refine the perturbation direction by attenuating its deterministic component.

At iteration t, we maintain a running estimator of the historical gradient direction:

$$\mathbf{m}_t = \beta \mathbf{m}_{t-1} + (1 - \beta) \nabla_{\omega} L_{tra}^{(t)}(\omega, \alpha), \tag{7}$$

where $\beta \in [0, 1)$ controls the smoothness of the estimate and $\nabla_{\omega} L_{tra}^{(t)}(\omega, \alpha)$ denotes the mini-batch gradient at iteration t. This moving average serves as a proxy for the dominant deterministic gradient component accumulated over training.

We then define a noise-consistent perturbation direction as:. The perturbation direction at iteration t is then defined as

$$\mathbf{d}_t = \nabla_\omega L_{tra}^{(t)}(\omega, \alpha) - \eta \mathbf{m}_t, \tag{8}$$

where $\eta \geq 0$ modulates the extent to which deterministic bias is suppressed. In practice, η is treated as a fixed scalar shared across training. The adversarial perturbation is subsequently computed as:

$$\epsilon_t = \rho \frac{\mathbf{d}_t}{\|\mathbf{d}_t\|_2}. \tag{9}$$

where ρ is the neighborhood radius introduced in III-B. Compared to standard SAM, this perturbation construction selectively amplifies batch-specific stochastic variations while reducing alignment with the global descent direction, yielding a perturbation that is more representative of local sharpness under stochastic training dynamics.

Based on the noise-consistent perturbation construction described above, we now formalize the resulting optimization objective used in the inner loop of GFNAS.

Recall that the stochastic training gradient at iteration t is denoted as:

$$g_t = \nabla_\omega L_{\text{tra}} \left(\omega, \alpha; B_{\text{tra}}^t\right), \tag{10}$$

and the EMA estimator of historical gradients is updated as:

$$m_t = \beta m_{t-1} + (1-\beta) g_t. \tag{11}$$

The noise-consistent perturbation direction is defined by suppressing the deterministic component estimated by m_t:

$$d_t = g_t - \eta m_t. \tag{12}$$

Substituting this perturbation into the sharpness-aware objective, the inner-loop optimization problem for a given architecture α can be written as:

$$\omega^*(\alpha) = \arg\min_\omega \mathbb{E}_{B^{\text{tra}}} \left[L_{\text{tra}} \left(\omega + \rho \frac{\Theta}{\|\Theta\|_2}, \alpha \right) \right] \tag{13}$$

where $\Theta = \nabla_\omega L_{\text{tra}} \ (\omega, \alpha; B^{\text{tra}}) - \eta m$ and the expectation is taken over mini-batches sampled from the training dataset.

IV. Experiments

A. Experiment Setup

1) Datasets: We adopt CIFAR-10 and CIFAR-100 [11], [17], together with their corrupted variants, as evaluation benchmarks because they are widely used and well established for robustness assessment. Both datasets contain 32×32 natural images with 50,000 training and 10,000 test samples, covering 10 and 100 classes, respectively, with balanced class distributions. Their compact scale and standardized splits make them particularly suitable for NAS under limited computational budgets. To evaluate robustness beyond clean accuracy, we further test on CIFAR-10-C and CIFAR-100-C, which introduce 15 types of synthetic corruptions, including noise, blur, weather effects, and digital artifacts. Models are trained on clean training sets and evaluated on both clean and corrupted data. Since the corrupted benchmarks are only provided at 32×32 resolution, we regenerate corrupted versions for each evaluated input resolution, as simple resizing would not preserve equivalent distortions.

B. Implementation Details

1) Experiment Environment: All experiments are implemented in PyTorch and conducted on a workstation equipped with NVIDIA A100 GPUs (80GB memory). During architecture search, we adopt a standard DARTS-style cell-based search space and perform the search on CIFAR-10 and CIFAR-100.

2) Optimization Settings: The shared network weights are optimized using stochastic gradient descent (SGD) with a momentum of 0.9, weight decay of 1×10^{-4}, and an initial learning rate of 0.025, following common practice in differentiable NAS. The architecture parameters are updated using the Adam optimizer with learning rate 1×10^{-4} [18], [24]. We apply basic data preprocessing and augmentation [26].

3) Sharpness-Aware Parameters Settings: For sharpness-aware training, the neighborhood radius is fixed to $\rho = 0.1$ throughout the search process. The exponential moving average (EMA) factor in gradient estimation is set to $\beta = 0.9$, and the deterministic gradient suppression coefficient is set to $\eta = 1.0$ in all experiments [28]. These hyperparameters are shared across datasets and architectures unless otherwise stated.

C. Comparison with State-Of-The-Art

1) Results on CIFAR-10: We compare GFNAS with representative NAS methods, including DARTS, RNAS, NADAR, RACL, and FlatNAS, under the same DARTS-style search space and retraining protocol. The results are reported in Table I. GFNAS achieves the highest Top-1 accuracy on the clean CIFAR-10 test set, outperforming all baselines and slightly improving upon FlatNAS. More importantly, GFNAS attains the lowest mean corruption error on CIFAR-10-C, indicating improved robustness under distribution shifts. Compared with FlatNAS, which also adopts sharpness-aware evaluation, this result suggests that noise-consistent sharpness estimation provides a more reliable robustness proxy during architecture search.

In addition, GFNAS maintains a compact model size and requires the lowest search cost, compared to 1.2 GPU days for FlatNAS and higher costs for other methods. These results demonstrate that GFNAS achieves a better balance among accuracy, robustness, and search efficiency on CIFAR-10.

TABLE I: Comparison with state-of-the-art NAS methods on CIFAR-10. Test Acc denotes Top-1 accuracy on the clean test set, mCE is the mean corruption error on CIFAR-10-C, Param. indicates the number of model parameters and Search Cost is measured in GPU days.

Methods	Acc (%)	mCE	Param. (M)	Search Cost
DARTS [21]	95.23	10.2	3.6	2.0
RNAS [39]	95.41	11.5	3.4	1.5
NADAR [20]	95.19	10.8	3.5	1.8
RACL [5]	95.03	10.5	3.7	2.2
FlatNAS [9]	95.44	9.9	3.3	1.2
GFNAS (Ours)	**95.61**↑	**9.7**↓	**3.3**	**1.0**↓

2) Results on CIFAR-100: As reported in Table II , our method demonstrates a clear overall advantage over representative state-of-the-art NAS approaches on CIFAR-100. In terms of predictive performance, it consistently achieves higher classification accuracy together with lower classification error, suggesting improved generalization ability rather than mere optimization of a single metric. At the same time, the resulting architectures are more compact, indicating that the proposed search strategy is able to identify effective architectural configurations without relying on excessive parameterization. Moreover, these performance gains are obtained with a reduced search cost, reflecting higher search efficiency compared to existing methods. This indicates that the proposed framework can explore the architecture space more effectively under limited computational budgets, making it particularly suitable for practical NAS scenarios where both performance and efficiency are critical.

TABLE II: Comparison with state-of-the-art NAS methods on CIFAR-100. Test Acc denotes Top-1 accuracy on the clean test set, mCE is the mean corruption error on CIFAR-100-C, Param. indicates the number of model parameters and Search Cost is measured in GPU days.

Methods	Acc (%)	mCE	Param. (M)	Search Cost
DARTS [21]	75.16	28.6	3.8	2.5
RNAS [39]	74.87	29.3	3.6	2.0
NADAR [20]	74.97	28.9	3.7	2.2
RACL [5]	75.31	28.7	3.9	2.8
FlatNAS [9]	75.24	28.2	3.5	1.8
GFNAS (Ours)	**75.35**↑	**27.9**↓	**3.4**↓	**1.7**↓

D. Ablation Study

1) Effectiveness of ρ: We additionally examine the effect of the perturbation radius ρ on the performance of GFNAS. In our experiments, we consider four representative values, $\rho \in \{0.05, 0.1, 0.2, 0.3\}$, and evaluate the resulting architectures on CIFAR-10 using Test Accuracy and mCE. As shown in Fig. 1, the setting $\rho = 0.1$ consistently achieves the best overall performance in terms of both Test Accuracy and mCE. Meanwhile, we observe that moderate variations of ρ around this value do not lead to noticeable performance degradation. This suggests that the proposed perturbation strategy allows for a reasonable operating range of ρ, while $\rho = 0.1$ provides a favorable balance between performance and stability in practice. Based on this observation, we adopt $\rho = 0.1$ as the default configuration in all subsequent experiments.

2) Effectiveness of η: We further study the effect of the suppression coefficient η which controls the strength of deterministic gradient bias removal in the proposed perturbation direction. T, we vary η over a representative range from weak suppression to relatively strong suppression, while keeping all other hyperparameters fixed, and evaluate the resulting architectures on CIFAR-10 using Test Accuracy and mCE.

We evaluate a range of deterministic suppression strengths by varying $\eta \in \{0.25, 0.5, 0.75, 1.0, 1.25\}$. As shown in Fig. 2, increasing η from smaller values leads to a consistent improvement in both Test Accuracy and robustness performance. In particular, the configuration $\eta = 1.0$ achieves the best overall results, attaining the highest Test Accuracy and the lowest mCE among all tested settings. This indicates that an appropriate level of deterministic bias suppression is beneficial for producing a more reliable evaluation signal during architecture search. When η is further increased beyond this value, performance begins to slightly degrade, suggesting that overly aggressive suppression may weaken useful gradient information. Based on these observations, we adopt $\eta = 1.0$ as the default setting in all subsequent experiments, as it provides a favorable balance between effective bias suppression and stable optimization behavior.

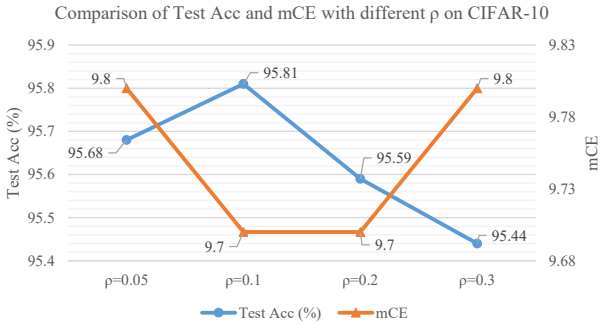

Fig. 1: Ablation study on the perturbation radius ρ on CIFAR-10. The setting $\rho = 0.1$ yields the best overall performance and is adopted as the default configuration.

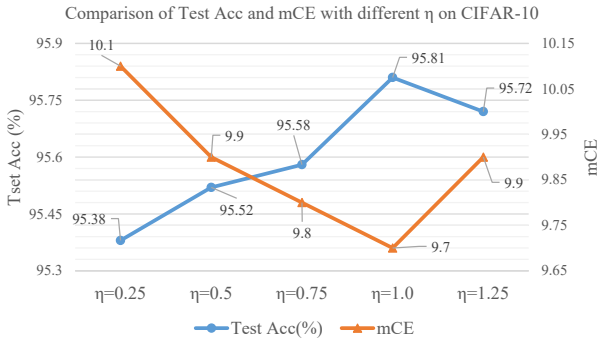

Fig. 2: Ablation study on the perturbation radius η on CIFAR-10. The setting $\eta = 1.0$ yields the best overall performance and is adopted as the default configuration.

V. Conclusion

In this paper, we revisited sharpness-aware neural architecture search from the perspective of gradient component decomposition and proposed GFNAS, a gradient-friendly sharpness-aware NAS framework that emphasizes noise-consistent sharpness signals while suppressing deterministic bias during the search process. By refining the perturbation construction in sharpness-aware optimization, GFNAS provides a more faithful robustness proxy for architecture evaluation without modifying the search space or increasing computational overhead. Extensive experiments on CIFAR benchmarks and their corrupted variants demonstrate that GFNAS consistently discovers architectures with improved generalization, robustness, and

reduced sensitivity to sharpness hyperparameters compared to existing sharpness-aware NAS methods. Overall, our results suggest that explicitly accounting for gradient component effects can improve the reliability of sharpness-based evaluation in NAS, and that noise-consistent sharpness estimation is a practical ingredient for robust and stable architecture search. In summary, GFNAS enables more robust architecture discovery with minimal changes to standard NAS pipelines.

REFERENCES

[1] Kayhan Behdin, Qingquan Song, Aman Gupta, David Durfee, Ayan Acharya, Sathiya Keerthi, and Rahul Mazumder. Improved deep neural network generalization using m-sharpness-aware minimization. *arXiv preprint arXiv:2212.04343*, 2022.

[2] Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In *International Conference on Learning Representations*, 2018.

[3] Hanning Chen, Lianbo Ma, Maowei He, Xingwei Wang, Xiaodan Liang, Liling Sun, and Min Huang. Artificial bee colony optimizer based on bee life-cycle for stationary and dynamic optimization. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2):327–346, 2017.

[4] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *Proceedings of the IEEE/CVF International Conference on computer vision*, pages 12239–12248, 2021.

[5] Minjing Dong, Yanxi Li, Yunhe Wang, and Chang Xu. Adversarially robust neural architectures. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (01):1–15, 2025.

[6] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21, 2019.

[7] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*.

[8] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv preprint arXiv:2010.01412*, 2020.

[9] Matteo Gambella, Fabrizio Pittorino, and Manuel Roveri. Flatnas: optimizing flatness in neural architecture search for out-of-distribution robustness. *arXiv preprint arXiv:2402.19102*, 2024.

[10] Hyeonjeong Ha, Minseon Kim, and Sung Ju Hwang. Generalizable lightweight proxy for robust nas against diverse perturbations. *Advances in Neural Information Processing Systems*, 36:38611–38623, 2023.

[11] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*.

[12] Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song, and Jacob Steinhardt. Pixmix: Dreamlike pictures comprehensively improve safety measures. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16783–16792, 2022.

[13] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 9(1):1–42, 1997.

[14] Joonhyun Jeong, Joonsang Yu, Dongyoon Han, and Y Yoo. Neural architecture search with loss flatness-aware measure. 2022.

[15] Weisen Jiang, Hansi Yang, Yu Zhang, and James Kwok. An adaptive policy to employ sharpness-aware minimization. *arXiv preprint arXiv:2304.14647*, 2023.

[16] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. *arXiv preprint arXiv:1912.02178*, 2019.

[17] Konstantin Kirchheim, Marco Filax, and Frank Ortmeier. Pytorch-ood: A library for out-of-distribution detection based on pytorch. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4351–4360, 2022.

[18] Bingcong Li and Georgios Giannakis. Enhancing sharpness-aware optimization through variance suppression. *Advances in Neural Information Processing Systems*, 36, 2024.

[19] Tao Li, Pan Zhou, Zhengbao He, Xinwen Cheng, and Xiaolin Huang. Friendly sharpness-aware minimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5631–5640, 2024.

[20] Yanxi Li, Zhaohui Yang, Yunhe Wang, and Chang Xu. Neural architecture dilation for adversarial robustness. *Advances in Neural Information Processing Systems*, 34:29578–29589, 2021.

[21] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations*.

[22] Hoang-Chau Luong and Minh-Triet Tran. Applying adaptive sharpness-aware minimization to improve out-of-distribution generalization. In *Proceedings of the 12th International Symposium on Information and Communication Technology*, pages 450–455, 2023.

[23] Lianbo Ma and Ying Liu. Maximizing three-hop influence spread in social networks using discrete comprehensive learning artificial bee colony optimizer. *Applied Soft Computing*, 83:105606, 2019.

[24] Peng Mi, Li Shen, Tianhe Ren, Yiyi Zhou, Xiaoshuai Sun, Rongrong Ji, and Dacheng Tao. Make sharpness-aware minimization stronger: A sparsified perturbation approach. *Advances in Neural Information Processing Systems*, 35:30950–30962, 2022.

[25] Xuefei Ning, Changcheng Tang, Wenshuo Li, Zixuan Zhou, Shuang Liang, Huazhong Yang, and Yu Wang. Evaluating efficient performance estimators of neural architectures. *Advances in Neural Information Processing Systems*, 34:12265–12277, 2021.

[26] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[27] Matt Poyser and Toby P Breckon. Neural architecture search: A contemporary literature review for computer vision applications. *Pattern Recognition*, 147:110052, 2024.

[28] Saul Shiffman. Ecological momentary assessment (ema) in studies of substance use. *Psychological assessment*, 21(4):486, 2009.

[29] Samuel L Smith and Quoc V Le. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018.

[30] Pengfei Wang, Zhaoxiang Zhang, Zhen Lei, and Lei Zhang. Sharpness-aware gradient matching for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3769–3778, 2023.

[31] Ruoyu Wang, Yuee Zhou, Hanning Chen, Lianbo Ma, and Meng Zheng. A surrogate-assisted many-objective evolutionary algorithm using multi-classification and coevolution for expensive optimization problems. *IEEE Access*, 9:159160–159174, 2021.

[32] Colin White, Arber Zela, Robin Ru, Yang Liu, and Frank Hutter. How powerful are performance predictors in neural architecture search? *Advances in neural information processing systems*, 34:28454–28469, 2021.

[33] Lingxi Xie, Xin Chen, Kaifeng Bi, Longhui Wei, Yuhui Xu, Lanfei Wang, Zhengsu Chen, An Xiao, Jianlong Chang, Xiaopeng Zhang, et al. Weight-sharing neural architecture search: A battle to shrink the optimization gap. *ACM Computing Surveys (CSUR)*, 54(9):1–37, 2021.

[34] Zeke Xie, Li Yuan, Zhanxing Zhu, and Masashi Sugiyama. Positive-negative momentum: Manipulating stochastic gradient noise to improve generalization. In *International Conference on Machine Learning*, pages 11448–11458. PMLR, 2021.

[35] Yan Xu, Jingwei Wang, Lianbo Ma, Junfeng Zhao, and Xiaolong Shen. Two-stage learning brain storm optimizer. In De-Shuang Huang, Vitoantonio Bevilacqua, and Abir Hussain, editors, *Intelligent Computing Theories and Application*, pages 28–40, Cham, 2020. Springer International Publishing.

[36] Antoine Yang, Pedro M Esperança, and Fabio M Carlucci. Nas evaluation is frustratingly hard. In *International Conference on Learning Representations*.

[37] Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. *arXiv preprint arXiv:1902.08142*, 2019.

[38] Ping Zhou, Jianhui Lv, Lianbo Ma, Yidan Chen, and Shujun Yang. An ant colony inspired cache allocation mechanism for heterogeneous information centric network. *IEEE Access*, 9:55485–55496, 2021.

[39] Xunyu Zhu, Jian Li, Yong Liu, and Weiping Wang. Robust neural architecture search. *arXiv preprint arXiv:2304.02845*, 2023.