# Supplementary Materials

## Generalizable Lightweight Proxy for Robust NAS against Diverse Perturbations

## A. Experimental Setting

**Search space.** Based on the cell-based neural architecture search space (Zoph et al., 2018), we regard the whole network as the composition of repeated cells. Thus, we search for the optimal cell architectures and stack them repeatedly to construct the entire network. In the cell-based search phase, each cell can be represented as a directed acyclic graph (DAG), which has $N$ nodes that represent the feature maps $z_j (j = 1, \cdots, N)$ and each edge between arbitrary node $i$ and node $j$ represents an operation $o_{i,j}$ chosen from the operation pool, where $o_{i,j} \in \mathcal{O} = \{o_k, k = 1, 2, \cdots, n\}$. Each feature map $z_j$ is obtained from all of its predecessors as follows:

$$x_j = \sum_{i<j} o_{i,j}(x_i) \tag{11}$$

In this work, we utilize NAS-Bench-201 (Dong & Yang, 2019), and DARTS (Liu et al., 2019) search space, where different operation pools are used, which are $\mathcal{O} = \{1 \times 1 \text{ conv.}, 3 \times 3 \text{ conv.}, 3 \times 3 \text{ avg. pooling, skip, zero}\}$, and $\mathcal{O} = \{3 \times 3 \text{ conv.}, 3 \times 3 \text{ dil.conv.}, 5 \times 5 \text{ conv.}, 5 \times 5 \text{ dil. conv.}, 7 \times 7 \text{ conv.}, 3 \times 3 \text{ max pooling}, 3 \times 3 \text{ avg. pooling, skip, zero}\}$, respectively. Especially, for the NAS-Bench-201 search space, we additionally use the Jung et al. (2023) dataset that includes robust accuracies of candidate neural architectures in NAS-Bench-201 search space to demonstrate the efficacy of our proposed proxy regarding searching generalized architectures against diverse perturbations and clean inputs.

**Baselines.** We consider three types of existing NAS approaches as our baselines. **1) Clean one-shot NAS** (Xu et al., 2020; Chen et al., 2020) One-shot NAS methods for searching architectures only on clean samples. **2) Clean zero-shot NAS** (Abdelfattah et al., 2021; Mellor et al., 2021): Zero-shot NAS with proxies that evaluate the clean performance of architectures without any training. **3) Robust NAS** (Guo et al., 2020; Mok et al., 2021) One-shot NAS methods for searching architectures only on adversarially perturbed samples.

**Standard Training & Adversarial Training.** For a fair comparison, we use the same training and testing settings to evaluate all the architectures searched with all NAS methods, including ours. 1) Standard Training: We train the neural architectures for 200 epochs under SGD optimizer with a learning rate of 0.1 and weight decay 1e-4, and use a batch size of 64 following (Mok et al., 2021). 2) Adversarial Training: We adversarially train the searched neural architectures with $l_\infty$ PGD attacks with the epsilon of 8/255, step size 2/255, and 7 steps. We evaluate the robustness against various perturbations, which are FGSM (Goodfellow et al., 2015), PGD (Madry et al., 2018), and common corruptions (Hendrycks & Dietterich, 2019). All attacks utilized in evaluation are $l_\infty$ attacks with the epsilon of 8/255.

**Adversarial Evaluation.** To evaluate standard-trained models, we utilize the robust NAS-Bench-201 (Jung et al., 2023) datasets, allowing us to achieve robust accuracy. On the other hand, to evaluate adversarially-trained models, we construct our own dataset, as described in Section 4.2, enabling us to obtain robust accuracy against adversarial attacks. In all our experiments, we obtain robust accuracy on CIFAR-10 against FGSM attack with an attack size ($\epsilon$) of 8.0/255.0 and attack step size ($\alpha$) of 8.0/2550.0 while we utilize robust accuracies against FGSM attack with attack size ($\epsilon$) of 4.0/255.0 on ImageNet16-120.

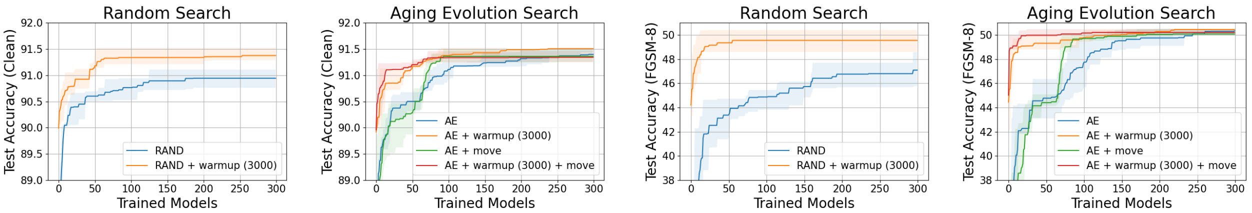
## B. Experimental Results

### B.1. End-to-End Generalization Performance on DARTS

**Efficient Sampling with CRoZe.** The efficiency of our proxy lies in its ability to evaluate neural architectures without iteratively training the models with those architectures. As a result, the majority of the search time is dedicated to the sampling of candidate architectures. To thoroughly explore the sampling costs involved in using our proxy to identify good neural architectures, we conduct experiments in the NAS-Bench-201 search space with two representative sampling-based search algorithms: random search (RAND) and evolutionary search (AE).

Our experiments show that CRoZe can rapidly identify an architecture with 91.5% clean accuracy and 50% robust accuracy

Figure 3: **Search with our proxy on CIFAR-10 in NAS-Bench-201 search space.** Our proxy can reduce the sampling costs in searching for neural architectures against both clean (left) and perturbed samples (right).



(a) Search on Clean Inputs (b) Search on Adversarially Perturbed Inputs

Table 4: Comparisons of the final performance and required computational resources of the searched network in DARTS search space on ImageNet16-120.

| NAS Method | NAS Training-free | Params (M) | # GPU | Batch Size | Standard-Trained | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | Clean | CC. | FGSM | HRS |
| PC-DARTS (Xu et al., 2020) | | 5.30 | 8 | 1024 | 50.58 | 14.36 | 0.15 | 0.30 |
| DrNAS (Chen et al., 2020) | | 5.70 | 8 | 512 | 49.63 | 13.42 | 0.21 | 0.42 |
| AdvRush (Mok et al., 2021) | | 4.20 | 1 | 64 | 38.72 | 10.39 | 0.11 | 0.22 |
| GradNorm (Abdelfattah et al., 2021) | ✓ | 5.90 | 1 | 8 | 39.13 | 10.75 | 0.23 | 0.46 |
| SynFlow (Abdelfattah et al., 2021) | ✓ | 6.13 | 1 | 8 | 43.73 | 12.10 | 0.15 | 0.30 |
| CRoZe | ✓ | 5.87 | 1 | 8 | 47.90 | 13.35 | 0.32 | 0.64 |

against FGSM attacks, even at the initial stages of the search. By leveraging the initial pool of architectures with high proxy values obtained from CRoZe (RAND+warmup and AE+warmup), our proxy effectively reduces the sampling costs and rapidly identifies high-performing architectures for clean inputs (Figure 3a). Moreover, by focusing sampling around the architecture with the highest proxy values of CRoZe (AE+warmup+move), we can search for architecture with even better robust performance using a less number of sampled models ($< 50$) (Figure 3b). Overall, our results highlight the effectiveness of CRoZe in searching for high-performing neural architectures against both clean and perturbed inputs in the NAS-Bench-201 search space.

**End-to-End Performance on ImageNet16-120.** We further validate the final performance of the searched neural architectures by CRoZe and compare the required computational resources with existing NAS frameworks including robust NAS (AdvRush (Mok et al., 2021)), clean one-shot NAS (PC-DARTS (Xu et al., 2020), DrNAS (Chen et al., 2020)) and clean zero-shot NAS (SynFlow and GradNorm (Abdelfattah et al., 2021)), on ImageNet16-120 in the DARTS search space. Similar to Section 4.3, we sample the same number (5,000) of candidate architectures using the warmup+move strategy for both clean-zero shot NAS and CRoZe.

The NAS Training-free methods such as GradNorm, SynFlow, and CRoZe only require a single GPU with a batch size of 8 to search for the architectures on the ImageNet16-120 dataset. In contrast, the existing clean one-shot NAS methods require 8 GPUs with much larger batch sizes. Moreover, NAS-Training-free methods consume less than 3000MB of memory, while both clean one-shot NAS and robust NAS need at least 3090 RTX GPU, which is available at 24000MB of memory. With its superior computational efficiency, CRoZe enables rapid neural architecture search and achieves the best HRS accuracy while maintaining comparable clean and common corruption accuracies. all at a much lower computational cost (Table 4). These demonstrate the effectiveness of CRoZe for rapid and lightweight robust NAS across diverse tasks (i.e., CIFAR-10, CIFAR-100, and ImageNet16-120) and perturbations (i.e., adversarial perturbations and common corruptions).

## B.2. Further Analysis

**Feature Variance of the Robust Models.** Our proxy searches for architectures that can generalize to all types of perturbed inputs, which are not overfitted to a single type of perturbation, as described in Section 3.1 (Eq. 2). To validate this, we analyzed the feature variance of neural architectures that demonstrated the highest performance against a single type of perturbation, namely FGSM and PGD, compared to an architecture selected with the CRoZe proxy from a pool of 300

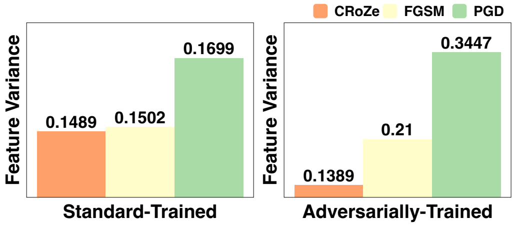


Figure 4: Feature variance across diverse perturbations.

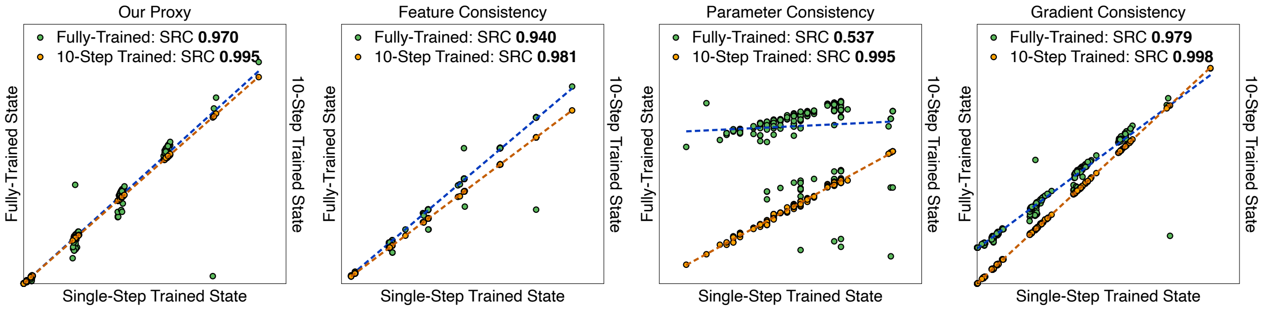


Figure 5: Spearman's $\rho$ in our proxy and each consistency component between the neural architectures with single-step trained states and two different states, which are fully-trained and 10-step trained states, respectively, against clean and perturbed images.

standard-trained and adversarially-trained models. Features were obtained from 18 different perturbations, including clean, PGD, FGSM, and 15 types of common corruptions. Remarkably, the neural architectures selected with our proxy exhibited the smallest standard deviations between the features obtained from the 18 different perturbations, in both standard training and adversarial training. Significantly, the architecture with the best PGD robustness demonstrated 2.48 times larger variations in the adversarial-training scenario, indicating a higher risk of overfitting to specific perturbations. These results provide compelling evidence of the effectiveness of CRoZe in identifying robust architectures that can consistently extract features and generalize across diverse perturbations.

**Generalization to Diverse Perturbations.** CRoZe evaluates the consistency between clean and perturbed inputs, regardless of the perturbation type, under the assumption that perturbed inputs retain the same semantic information as clean inputs. To empirically demonstrate the independence of our proxy from specific perturbation types, we introduce random Gaussian noise in place of adversarial perturbations in Eq. 4. As evidenced in Table 5, CRoZe consistently exhibits similar Spearman's $\rho$ in the NAS-Bench-201 search space. This emphasizes that CRoZe captures the characteristics of robust architectures through the consistency of clean and perturbed inputs, irrespective of the perturbation types employed.

| | Standard-Trained | | | | |
|---|---|---|---|---|---|
| | Clean | CC. | FGSM $(\epsilon, 8)$ | FGSM $(\epsilon, 4)$ | FGSM $(\epsilon, 2)$ |
| Gaussian Noise | 0.810 | 0.436 | 0.821 | 0.797 | 0.778 |
| Adversarial | 0.823 | 0.436 | 0.823 | 0.826 | 0.801 |
| | Adversarially-Trained | | | | |
| | Clean | PGD | HRS(PGD) | FGSM $(\epsilon, 8)$ | HRS(FGSM) |
| Gaussian Noise | 0.718 | 0.503 | 0.590 | 0.415 | 0.603 |
| Adversarial | 0.723 | 0.501 | 0.588 | 0.417 | 0.602 |

Table 5: Comparisons of perturbation type in CRoZe.

**Assessment of CRoZe Predictiveness.** To empirically validate whether our proxy obtained from a random state accurately represents the proxy in the fully-trained model, we conducted an analysis using Spearman's $\rho$. We randomly sampled 300 architectures from the NAS-Bench-201 search space and trained them on the entire dataset, using both the full training and reduced training only of 10 steps. The Spearman's $\rho$ between the proxy value obtained from a single-step trained state and the 10-step trained state shows a strong correlation of 0.995 (Figure 5). Even after full training, the correlation remained high at 0.970 in a standard training scenario. These suggest that our proxy, derived from estimated surrogate networks, can significantly reduce the computational costs associated with obtaining fully-trained models. Furthermore, each component of CRoZe consistently demonstrates a high correlation with the fully-trained states, supporting the notion that the high correlation computed with the final state is not solely a result of the collective influence of components, but rather an accurate estimation.

**Validation of CRoZe for Estimating Fully-Trained Neural Architectures.** To investigate whether the architectures selected with our proxy possess the desired robustness properties discussed in Section 3.1, we conduct a comprehensive analysis. From a pool of 300 samples in the NAS-Bench-201 search space, we select the top-5 and bottom-5 neural architectures based on our proxy values, excluding those with a robust accuracy of less than 20%. We evaluate the performances of these architectures that are standard-trained and adversarially-trained against clean inputs and three distinct types of perturbed inputs: adversarial perturbations (i.e., FGSM and PGD), and common corruptions. Notably, the top-5 networks show impressive average clean accuracies of 89.62%, and 84.36% on CIFAR-10, while the bottom-5 networks achieve significantly lower average clean accuracies of 61.34%, 51.64% in standard training and adversarial training, respectively (Table 6).

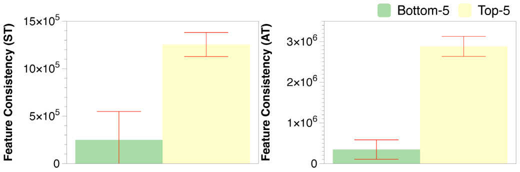


Figure 6: Comparisons of feature consistency in fully-trained states.

Table 7: Comparison of Spearman's $\rho$ between the actual accuracies and the proxy values on CIFAR-10 in the NAS-Bench-201 search space. Clean stands for clean accuracy and robust accuracies are evaluated against adversarial perturbations (FGSM (Goodfellow et al., 2015)) and common corruptions (CC. (Hendrycks & Dietterich, 2019)). Avg. stands for average Spearman's $\rho$ values with all accuracies.

| Proxy components | | | Standard-Trained | | | |
|---|---|---|---|---|---|---|
| Feature | Parameter | Gradient | Clean | FGSM | CC. | Avg. |
| ✓ | – | – | 0.718 | 0.701 | 0.341 | 0.587 |
| ✓ | ✓ | – | 0.750 | 0.762 | 0.384 | 0.632 |
| ✓ | – | ✓ | 0.822 | 0.824 | 0.434 | 0.693 |
| – | ✓ | ✓ | 0.824 | 0.827 | 0.437 | 0.696 |
| ✓ | ✓ | ✓ | 0.823 | 0.826 | 0.436 | 0.695 |

(a) Standard-Trained

| Proxy components | | | Adversarially-Trained | | | | |
|---|---|---|---|---|---|---|---|
| Feature | Parameter | Gradient | Clean | FGSM | PGD | HRS(FGSM) | HRS(PGD) |
| ✓ | – | – | 0.602 | 0.295 | 0.329 | 0.442 | 0.404 |
| ✓ | ✓ | – | 0.677 | 0.343 | 0.431 | 0.542 | 0.527 |
| ✓ | – | ✓ | 0.707 | 0.405 | 0.489 | 0.587 | 0.573 |
| – | ✓ | ✓ | 0.731 | 0.422 | 0.507 | 0.610 | 0.595 |
| ✓ | ✓ | ✓ | 0.723 | 0.417 | 0.501 | 0.602 | 0.588 |

(b) Adversarially-Trained

Moreover, the top-5 networks exhibit an average robust accuracy of 54.95% against perturbations, compared to only 28.50% for the bottom-5 group. Furthermore, by considering feature consistency between clean and perturbed inputs, as defined in Eq. 2, the top-5 group exhibited similar features, whereas the bottom-5 group showed dissimilar features (Figure 6). This supports that architectures with consistent features across inputs are more likely to make accurate predictions under perturbations. Overall, our analysis provides strong evidence that our proxy can search for generalizable robust architectures that accurately predict the final accuracies under diverse perturbations.

| | Standard-Trained | | | | |
|---|---|---|---|---|---|
| | Clean | CC. | PGD | FGSM | Avg. Rob. |
| Top-5 | 89.62 | 77.68 | 26.09 | 61.08 | 54.95 |
| Bottom-5 | 61.34 | 49.13 | 19.35 | 17.01 | 28.50 |
| | Adversarially-Trained | | | | |
| | Clean | PGD | HRS(PGD) | FGSM | HRS(FGSM) |
| Top-5 | 84.36 | 40.55 | 54.76 | 66.48 | 74.34 |
| Bottom-5 | 51.64 | 22.59 | 31.36 | 37.20 | 43.04 |

Table 6: Comparisons of the performance.

**Ablation on Each Component of CRoZe.** In Section 3.3, we introduce our proxy, which consists of three components: feature consistency, parameter similarity, and gradient similarity. We discuss the importance of considering all three components to accurately evaluate the robustness of neural architectures in a random state (Section 3.1). To analyze the contributions of each factor, we conduct an ablation study in both the NAS-Bench-201 search space and the DARTS search space.

We find that relying solely on feature consistency in a random state is insufficient to evaluate the robustness of architectures. The proxy with only feature consistency shows a lower correlation in both standard training and adversarial training scenarios compared to CRoZe in the NAS-Bench-201 search space (Table 7a and Table 7b). This indicates that high scores obtained based on feature consistency on a single batch may not accurately reflect the performance across the entire dataset. On the other hand, when parameter or gradient similarity is added to the proxy, the correlation consistently improved, suggesting that these factors complement feature consistency by imposing stricter constraints on the parameter space and convergence directions, respectively. While the proxy considering only parameter and gradient similarity achieves better Spearman's $\rho$ compared to our proxy, the top-3 architectures chosen by our proxy exhibit higher average performance than those discovered by the proxy without feature consistency (Table 8a). Moreover, CRoZe selects the same top-1 candidate architecture as the proxy without feature consistency in the NAS-Bench-201 search space.

We further conduct ablation experiments on CIFAR-10 in the DARTS search space, which contains about $10^{19}$ number of candidate architectures that is significantly larger than the NAS-Bench-201 search space containing 15625 architectures. The proxy without feature consistency yielded architectures with poor robust accuracies, while the architectures selected by our proxy consistently outperformed the former on both clean and perturbed images (Table 8b). Furthermore, the average

Table 8: Comparisons of the final performance of the searched network in NAS-Bench-201 and DARTS search space on CIFAR-10. **Bold** and underline stands for the best and second.

| Proxy components | | | Standard-Trained (Top-1) | | | | Standard-Trained (Top-3) | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Feature | Parameter | Gradient | Clean | FGSM | CC. | Avg. | Clean | FGSM | CC. | Avg. |
| ✓ | – | – | 93.30 | 44.30 | 55.62 | 64.41 | 92.93 | 37.60 | 54.03 | 61.52 |
| ✓ | ✓ | – | 93.70 | 45.80 | 56.93 | 65.48 | 93.63 | 48.20 | 55.39 | **65.74** |
| ✓ | – | ✓ | 93.70 | 45.80 | 56.93 | 65.48 | 93.63 | 48.20 | 55.39 | **65.74** |
| – | ✓ | ✓ | 93.70 | 45.80 | 56.93 | 65.48 | 93.43 | 41.37 | 55.30 | 63.37 |
| ✓ | ✓ | ✓ | 93.70 | 45.80 | 56.93 | 65.48 | 93.93 | 43.87 | 56.11 | 64.64 |

(a) NAS-Bench-201 search space

| Proxy Components | | | Standard-Trained | | | | |
|---|---|---|---|---|---|---|---|
| Feature | Parameter | Gradient | Clean | CC. | FGSM | HRS | Avg. |
| ✓ | – | – | 94.37 | 72.26 | 16.87 | 28.62 | 53.78 |
| ✓ | ✓ | – | **94.99** | 74.06 | 16.82 | 28.58 | 53.86 |
| ✓ | – | ✓ | 94.30 | **74.91** | 16.67 | 28.33 | 53.55 |
| – | ✓ | ✓ | 94.34 | 74.46 | 15.71 | 26.93 | 52.61 |
| ✓ | ✓ | ✓ | 94.45 | 74.63 | **22.38** | **33.19** | **56.41** |

(b) DARTS search space

performance of the architectures chosen by the proxy based solely on feature consistency shows even higher than that of the architectures discovered by the proxy without considering feature consistency. This clearly demonstrates the influential role of feature consistency in evaluating robustness. Overall, our proposed proxy effectively searches high-performing architectures by employing consistency across features, parameters, and gradients to estimate the robustness of the given architectures within a single gradient step. The overall algorithm of CRoZe is described in Algorithm 1.

**Algorithm 1** **C**onsistency-based **Ro**bust **Ze**ro-cost Proxy (**CRoZe**).

```
Input: A single batch of given dataset $B = \{(x, y)\}$
Input: Network $f_\theta(\cdot)$ consists of $M$ layer, which is architecture $\mathcal{A}$ with parameterized by $\theta$
Estimate robust network $f_{\theta^r}$ as done in Eq. 3.
for $m = 1, \cdots, M$ do
    $\theta^r_m \leftarrow \theta_m + \beta * \frac{\nabla_{\theta_m}\mathcal{L}\big(f_\theta(x), y\big)}{\|\nabla_{\theta_m}\mathcal{L}\big(f_\theta(x), y\big)\|} * \|\theta_m\|$
end for
Generate perturbed input $x'$ using $f_{\theta^r}$. as done in Eq. 4.
$\delta = \epsilon \texttt{sign}\Big(\nabla_x \mathcal{L}(f_{\theta^r}(x), y)\Big)$
$x' = x + \delta$
Calculate consistency in features, parameters, and gradients as done in Section 3.3.
$g = \nabla_\theta \mathcal{L}\big(f_\theta(x), y\big)$
$g^r = \nabla_{\theta^r} \mathcal{L}\big(f_{\theta^r}(x'), y\big)$
Single gradient step for clean network $f_\theta$ and robust network $f_{\theta^r}$
$\theta_1 \leftarrow \theta - \gamma g$
$\theta^r_1 \leftarrow \theta^r - \gamma g^r$
for $m = 1, \cdots, M$ do
    Feature consistency
    $\mathcal{Z}_m(f_\theta(x), f_{\theta^r}(x')) = 1 + \frac{z_m \cdot z^r_m}{\|z_m\| \|z^r_m\|}$
    Parameter consistency
    $\mathcal{P}_m(\theta_1, \theta^r_1) = 1 + \frac{\theta_{1,m} \cdot \theta^r_{1,m}}{\|\theta_{1,m}\| \|\theta^r_{1,m}\|}$
    Gradient consistency
    $\mathcal{G}_m(g, g^r) = \Big|\frac{g_m \cdot g^r_m}{\|g_m\| \|g^r_m\|}\Big|$
end for
CRoZe = $\sum_{m=1}^{M} \mathcal{Z}_m \times \mathcal{P}_m \times \mathcal{G}_m$
```