Unleashing the Power of Gradient Signal-to-Noise Ratio for Zero-Shot NAS

Zihao Sun\textsuperscript{1,3} \textsuperscript{†}, Yu Sun\textsuperscript{2,3} \textsuperscript{†}, Longxing Yang\textsuperscript{1,3}, Shun Lu\textsuperscript{1,3}, Jilin Mei\textsuperscript{1}, Wenxiao Zhao\textsuperscript{2,3} \textsuperscript{*}, Yu Hu\textsuperscript{1,3} \textsuperscript{*}

\textsuperscript{1}Research Center for Intelligent Computing Systems, Institute of Computing Technology, Chinese Academy of Sciences
\textsuperscript{2}Key Laboratory of Systems and Control, Academy of Mathematics and Systems Science, Chinese Academy of Sciences
\textsuperscript{3}University of Chinese Academy of Sciences

\{sunzhao18z, yanglongxing20b, lushun19s, meijilin, huyu\}@ict.ac.cn, \{sunyu211\}@mails.ucas.ac.cn, \{wxzhao\}@amss.ac.cn

Abstract

Neural Architecture Search (NAS) aims to automatically find optimal neural network architectures in an efficient way. Zero-Shot NAS is a promising technique that leverages proxies to predict the accuracy of candidate architectures without any training. However, we have observed that most existing proxies do not consistently perform well across different search spaces, and are less concerned with generalization. Recently, the gradient signal-to-noise ratio (GSNR) was shown to be correlated with neural network generalization performance. In this paper, we not only explicitly give the probability that larger GSNR at network initialization can ensure better generalization, but also theoretically prove that GSNR can ensure better convergence. Then we design the \(\xi\)-based gradient signal-to-noise ratio (\(\xi\)-GSNR) as a Zero-Shot NAS proxy to predict the network accuracy at initialization. Extensive experiments in different search spaces demonstrate that \(\xi\)-GSNR provides superior ranking consistency compared to previous proxies. Moreover, \(\xi\)-GSNR-based Zero-Shot NAS also achieves outstanding performance when directly searching for the optimal architecture in various search spaces and datasets. The source code is available at https://github.com/Sunzh1996/Xi-GSNR.

1. Introduction

Neural architecture search (NAS) \cite{16} is a technique that automates the design of neural network architectures, easing the burden of human trial and error \cite{19, 44}. The main challenge of NAS is to evaluate the performance of each candidate architecture in a given search space. Early approaches that trained each architecture separately until convergence were time-consuming and resource-intensive\cite{3, 68, 42, 41}. To improve the search efficiency, ENAS \cite{38} proposed to share the same operation weights in a super-network. This reduces the training cost to only one super-network, namely One-Shot NAS \cite{4]. Gradient-based NAS \cite{32, 64, 8, 57, 14, 11, 9, 58} and Sampling-based NAS \cite{17, 10, 61, 65, 62} are two mainstream paradigms for training super-networks. However, they still require training before evaluating each candidate architecture, suffering from extra search overhead.

In order to eliminate the need for training in NAS, Zero-Shot NAS \cite{35} was developed to evaluate network performance without any training, thus significantly promoting search speed. To achieve this, a series of proxies are designed to predict the performance of a given candidate architecture. Some of these proxies are based on empirical inspiration from the pruning-at-initialization literature, using saliency metrics such as SNIP \cite{23}, GraSP \cite{54}, SynFlow \cite{51} and so on. Whereas, recent studies \cite{1, 36} have shown that these proxies perform poorly in correlating with network performance. Other Zero-Shot proxies are theoretically designed from the deep learning theory of neu-
ral networks [35, 30, 46, 7, 66, 28]. For example, some proxies [35, 30] analyzed the relationship between the linear region [40] and network performance, some leveraged Neural Tangent Kernel (NTK) [21] to assess the predictive performance of a network, while others [66, 28] analyzed the gradients of the sample-wise optimization [2]. Nevertheless, most existing proxies cannot consistently perform well across different search spaces, and are less concerned with generalization, motivating us to address this issue.

As revealed in recent studies [47, 53], architectures with faster convergence are preferred by NAS algorithms, but may not guarantee good generalization performance. This insight motivates us to consider both generalization and convergence when designing proxy indicators. The gradient signal-to-noise ratio (GSNR) was proposed in [33], and is defined as the ratio between the squared mean and variance of the parameter gradient. In this paper, we clearly give the probability guarantee of good generalization is positively correlated with higher GSNR. Moreover, we have further provided theoretical evidence of a strict correlation between larger GSNR and better network convergence. To verify our theory, we conduct a toy experiment in NAS-Bench-201 on CIFAR-10. We randomly sample 800 architectures and train them for one epoch. As illustrated in Fig.1, GSNR is positively correlated with the generalization ratio and negatively correlated with the training loss. This result suggests that GSNR can effectively capture the generalization and convergence at network initialization.

In this paper, we further develop a novel Zero-Shot NAS proxy called ξ-based gradient signal-to-noise ratio (ξ-GSNR), which introduces a fairly small ξ term to smooth the variance of the parameter gradient. The network performance is then predicted by ξ-GSNR proxy at network initialization without any training. A larger ξ-GSNR value indicates better network performance. We conduct experiments in various NAS-Benches and observe that the ξ-GSNR proxy exhibits significant ranking consistency with network performance, surpassing existing Zero-Shot NAS proxies. In addition, when directly searching for architectures in different search spaces, the accuracy has also been boosted due to the effectiveness of ξ-GSNR proxy.

To this end, we summarize our contributions as follows:

• We provide theoretical proof that gradient signal-to-noise ratio (GSNR), which is the ratio between the squared mean and variance of the parameter gradient, is positively correlated with the generalization and convergence of the network at initialization.

• We develop a novel Zero-Shot NAS proxy called ξ-based gradient signal-to-noise ratio (ξ-GSNR) by introducing a fairly small ξ term to smooth the variance of the parameter gradient. ξ-GSNR is more accurate than vanilla GSNR in predicting network performance.

• Our experimental results indicate that ξ-GSNR proxy achieves significant ranking consistency with network accuracy in various NAS-Benches. In addition, the performance of the searched architecture has also been improved when involving ξ-GSNR proxy during the searching procedure of NAS.

2. Related Works

We reviewed related works of One-Shot NAS and Zero-Shot NAS from the perspective of performance evaluation.

One-Shot NAS. To address the high search overhead of traditional heuristic methods [3, 68, 42, 41], which typically require training each architecture to convergence, One-Shot NAS [4] presents a simpler alternative. This approach involves training only one super-network, allowing all sub-networks can be evaluated through weight-sharing [38]. DARTS [32] and SPOS [17] thereafter became two popular paradigms for super-network training. Specifically, DARTS and its variants [6] and [11, 9, 58] alternately optimize architecture parameters and network weights. The final architecture is derived from the optimal architecture parameters on each edge in a cell. Path sampling-based approaches [10, 61, 65, 62] focus on ensuring the fairness and robustness of super-network training. In this way, the performance of sub-networks is evaluated by inheriting the super-network weights. Despite alleviating the cost of evaluating candidate architectures, One-Shot NAS still bears the burden of super-network training.

Zero-Shot NAS. To completely liberate NAS from training, Zero-Shot NAS [35] was proposed to evaluate the network performance without any training. By designing a series of Zero-Shot NAS proxies to estimate the performance of candidate architectures at initialization, the search speed has been significantly promoted. We broadly categorize existing proxies as empirically designed or theoretically designed ones. On the one hand, some Zero-Shot NAS proxies are designed to identify the saliency metric from the perspective of pruning-at-initialization, including SNIP [23], GraSP [54], SynFlow [51] and so on. However, some studies [1, 36] have shown that these training-free proxies are not robust in ranking candidate architectures across different search spaces. Moreover, the simplest metric, such as the number of parameters or FLOPs, can outperform empirically inspired proxies [1, 36].

On the other hand, some Zero-Shot NAS proxies are inspired by the deep learning theory of neural networks. For instance, linear region analysis [40] helps to assess network predictive performance. Different from NASWOT [35], Zen-NAS [30] defined the expected Gaussian complexity to measure network expressivity. As another example, Neural Tangent Kernel (NTK) [21] discusses the training dynamics of infinite-width neural network. NASI [46] leveraged the capability of Neural Tangent Kernel (NTK) to characterize
network performance at initialization. TE-NAS [7] traded off the theory of Neural Tangent Kernel (NTK) and the linear regions of networks to evaluate its trainability and expressivity. NNPG [37] approximated network performance by fitting the kernel regression parameters. GradSign [66] and ZiCo [28] analyzed the sample-wise gradient optimization of a network to quantify its performance. However, as illustrated in [47, 53] that the architecture with faster convergence does not necessarily generalize better, indicating that both generalization and convergence are crucial in characterizing network performance. This motivates us to design a novel Zero-Shot NAS proxy, taking into account of generalization and convergence simultaneously.

3. Methodology

We first give the preliminary of gradient signal-to-noise ratio (GSNR) illustrated in [33] (Sec.3.1). Then we analyze the property of GSNR (Sec.3.2). We further provide theoretical evidence that GSNR has a strong correlation with network generalization (Sec.3.3) and convergence (Sec.3.4). Finally, to enhance the performance, we develop a novel Zero-Shot NAS proxy named ξ-GSNR (Sec.3.5).

3.1. Preliminary

Let \( \theta \in \mathbb{R}^p \) represent the parameters of the neural network \( f(x, \theta) \) function, \( \theta_j \) denotes the \( j \)-th parameter \( \theta \). Dataset \( D = \{(x_i, y_i), i = 1, ..., n\}, (x_i, y_i) \sim \mathcal{D}. \) \( \theta \) is optimized with loss function \( \mathcal{L} \) via gradient descent:

\[
g_D(\theta_j) = \frac{1}{n} \sum_{i=1}^{n} g(x_i, y_i, \theta_j) := \frac{1}{n} \sum_{i=1}^{n} \frac{\partial \mathcal{L}(y_i, f(x_i, \theta))}{\partial \theta_j}
\]

(1)

The gradient signal-to-noise ratio (GSNR) of one parameter \( \theta_j \) is defined as:

\[
\text{GSNR}(\theta_j) := \frac{(\mathbb{E}_{(x,y)\sim \mathcal{D}} g(x, y, \theta_j))^2}{\text{Var}_{(x,y)\sim \mathcal{D}} g(x, y, \theta_j)}
\]

(2)

\( \text{GSNR}(\theta_j) \) measures the similarity of parameter gradients across different training samples. Large \( \text{GSNR}(\theta_j) \) implies that the parameter gradients tend to be in the similar direction, leading to better generalization and convergence.

3.2. Property of GSNR

To facilitate the proof, we introduce the gradient signal-to-noise ratio of parameter \( \theta_j \) via the output of network:

\[
gsnr(\theta_j) := \frac{(\mathbb{E}_{(x,y)\sim \mathcal{D}} g'(x, \theta_j))^2}{\text{Var}_{(x,y)\sim \mathcal{D}} g'(x, \theta_j)}
\]

(3)

where \( g'(x, \theta_j) = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial f(x, \theta)}{\partial \theta_j} \). Next, we deduce the correlation between \( \text{GSNR}(\theta_j) \) and \( gsnr(\theta_j) \) is positive, and thus can associate network generalization and convergence with \( gsnr(\theta_j) \) at initialization.

**Theorem 1** \( \forall 1 > \epsilon > 0, j = 1, ..., P, \exists M_1 \text{ such that with probability at least } (1 - \epsilon) \text{ over randomly initialized parameters } \theta^0, \)

\[
\frac{M_1}{16M^2 - M_1} \leq gsnr(\theta_j^0)
\]

(4)

where \( M \) is the upper bound of the output.

The proof details are provided in Appendix A.1. Theorem 1 shows that with probability arbitrarily close to 1 over random network initialization, the larger \( \text{GSNR}(\theta_j^0) \), the larger \( gsnr(\theta_j^0) \) is. In the following sections, we will prove the generalization and convergence effect brought by the increase of \( gsnr(\theta_j^0) \), and thus associate it with \( \text{GSNR}(\theta_j^0) \).

3.3. Generalization Analysis

As illustrated in [33] that at each step of gradient descent, the larger \( \text{GSNR} \) implies a greater proximity between the expected reductions in validation loss and training loss. Unfortunately, this does not indicate the probability that the loss reduction of any given validation set will be similar to that of the training set. After several steps of gradient descent, the accumulation of bias can lead to substantial discrepancies between the training loss and validation loss. As a result, we provide a novel and comprehensive proof of the generalization guarantee.

The maximum eigenvalue of the training loss Hessian matrix is a measurement of the flatness of the loss landscape, which is highly correlated with the generalization ability of a neural network [43, 24, 22, 12]. When the minimum point of validation loss is similar to that of training loss, both losses are flat, it can ensure that the difference between the two losses is minimal (see Fig.2(a)). On
the other hand, if the validation loss is sharp, the difference between the two losses will be significant, resulting in poor generalization (see Fig.2(b)). Next, we will prove that when GSNR is larger, it can ensure with a higher probability that the Hessian maximum eigenvalue of training loss is upper bounded by that of validation loss.

**Assumption 1** $gsnr(\theta_j)$ and $E_{D \sim Z^n} \nabla_\theta f_D(\theta)$ are continuous functions and robust, which means that there is a neighborhood around fixed initialization parameters $\theta^0$ such that for any two parameters $\alpha$ and $\beta$, $|gsnr(\alpha_j) - gsnr(\beta_j)|$, $j = 1, 2...P$ and $\|E_{D \sim Z^n} \nabla_\theta f_D(\alpha) - E_{D \sim Z^n} \nabla_\theta f_D(\beta)\|$ are small.

This assumption implies that after gradient descent, $gsnr(\theta^0)$ and $gsnr(\theta^t_j)$ are close. Therefore, we can analyze the generalization and convergence using $gsnr(\theta^t_j)$ to approximate that of $gsnr(\theta^0_j)$.

**Theorem 2** Under Assumption 1, for fixed initialization parameters $\theta^0$, if $\nabla^2_D L^t(\theta^t)$ is semi-positive definite matrix, $E_{(x,y) \sim Z}[(f(x, \theta^t) - y)]$ is small enough, $\forall t = 1, 2, ..., \exists 0 < \alpha_t < 1$ and $\frac{1}{\alpha_t gsnr(\theta^0_j)} < r < 1$, $j = 1, 2...P$, such that

$$\lambda_{\max}(\nabla^2_D L^t(\theta^t)) \leq \frac{n(1 + r)^2}{(1 - r)^2} \lambda_{\max}(\nabla^2_D L^t(\theta^t))$$

with probability at least

$$1 - \sum_{j=1}^P \frac{2n}{r^2 \alpha_t gsnr(\theta^0_j)}$$

over randomly chosen possible distributions for all training datasets $D$ and validation datasets $D'$ which have the same number of data. $\lambda_{\max}(\cdot)$ means the maximum eigenvalue.

The proof details are provided in Appendix A.2. Theorem 2 states that when $gsnr(\theta^0_j)$ is larger, the probability that the Hessian maximum eigenvalue of validation loss bounded by that of training loss will close to 1. This implies that both the training loss and validation loss are flat, just like Fig.2(a), indicating a better generalization. Combining Theorem 1, we can conclude that larger $GSNR(\theta^0_j)$ will result in better generalization performance.

### 3.4. Convergence Analysis

We prove the convergence guarantee from the perspective of training loss reduction, i.e., the greater the loss reduction between each two steps, the faster the convergence. We give the following theorem:

**Theorem 3** Under Assumption 1, for fixed initialization parameters $\theta^0$, if the learning rate $\eta$ is small enough, $\forall t = 1, 2..., \exists 0 < \alpha_t < 1$ and $\frac{1}{\max_{j} gsnr(\theta^0_j)} < r < 1$, $j = 1, 2...P$, such that,

$$L_D(\theta^{t+1}) - L_D(\theta^t)$$

$$< -\eta \alpha_t (1 - r)^2 (\frac{\partial L_D}{\partial \theta^t}(\theta^t))^2 E_{D \sim Z^n}(\sum_{j=1}^P (g_D(\theta^0_j))^2)$$

with probability at least

$$1 - \sum_{j=1}^P \frac{1}{\eta^{2 \alpha_t} gsnr(\theta^0_j)}$$

over randomly chosen possible distributions for all training datasets $D$.

The proof details are provided in Appendix A.3. Theorem 3 shows that larger $gsnr(\theta^0_j)$ significantly increases the likelihood of Eq.(7) approaching 1. Besides, $E_{D \sim Z^n}(\sum_{j=1}^P (g_D(\theta^0_j))^2)$ is positively correlated with $gsnr(\theta^0_j)$, which indicates that larger $gsnr(\theta^0_j)$ leads to faster convergence rate. Combining Theorem 1, we can conclude that larger $GSNR(\theta^0_j)$ is advantageous in accelerating the convergence speed of neural networks.

### 3.5. Zero-Shot NAS Proxy

In the vanilla definition of $GSNR$ in Eq.(2), the extremely small gradient variance of a parameter $\theta_j$ may dominate the summation of all parameters gradient signal-to-noise ratio, resulting in an improper proxy score. Therefore, we enhance it as $\xi$-GSNR by introducing a small $\xi$ term to smooth the gradient variance:

$$\xi$-GSNR := \sum_{j=1}^P (E_{(x,y) \sim Z}(g(x,y,\theta_j))^2 \sqrt{Var_{(x,y) \sim Z}(g(x,y,\theta_j))} + \xi$$

In Eq.(9), we introduce $\xi$ which is set to be quite small for stabilizing the computation and producing a smooth proxy score. The ablation study further demonstrates the effectiveness of $\xi$ in improving the ranking consistency as shown in Tab.9.

In our practical experiment, we average a large number of samples (set to $M$) gradients to obtain an approximation of gradient expectations. To facilitate the calculation, we divide $M$ data samples into $N$ batches $\{B_i\}_{i=1}^N$. Thus, our practical Zero-Shot NAS proxy is denoted as:

$$\xi$-GSNR := \sum_{j=1}^P \left( \frac{1}{N} \sum_{i=1}^N g_B(\theta_j) \right)^2 - \left( \frac{1}{N} \sum_{i=1}^N g_B(\theta_j) \right)^2 + \xi$$

where $g_B(\theta_j) := \frac{1}{k} \sum_{i=1}^k \frac{\partial f_B(y_i, f(x_i, \theta_j))}{\partial \theta_j}$ denotes the gradient of batch data with $k$ sample data in the batch $B$. 

5766
4. Experiments

We first conduct the ranking consistency experiments (Sec.4.1) in various NAS-Benches to verify the effectiveness of ξ-GSNR proxy. Then, we introduce the ξ-GSNR proxy into Zero-Shot NAS (Sec.4.2) to directly search for the optimal architecture in different search spaces and datasets. Finally, the ablation studies (Sec.4.3) are analyzed in detail of ξ-GSNR proxy.

4.1. Ranking Consistency

Baselines. We compare our ξ-GSNR with existing popular Zero-Shot NAS proxies, including both empirically inspired and theoretically designed ones. Specifically, the former ones are SNIP [23], GraSP [54], SynFlow [51] and GradNorm [1]; while the latter ones consist of NASWOT [35], Zen-Score [30], TE-NAS [7], GradSign [66], and ZiCo [28]. We also evaluate the number of parameters and FLOPs, as they are strong baselines highlighted in recent work [36]. To ensure fairness, we keep the parameters initialization and batch size consistent across all proxies.

Metrics. Spearman’s ρ [18] and Kendall’s τ are two widely used metrics for characterizing the correlation between two sequences. We leverage them to quantify the relationship between Zero-Shot NAS proxies and architecture performance. The range of both ρ and τ is [-1, 1]. When the metric value approaches 1, it implies that the proxy is more reliable in predicting network performance.

4.1.1 Results in NAS-Bench-201

NAS-Bench-201 [15] is a comprehensive cell-based search space that includes almost up-to-date NAS algorithms. The Benchmark comprises 15,625 candidate architectures and provides access to the diagnostic information on accuracy, loss, and parameters across three datasets: CIFAR-10, CIFAR-100, and ImageNet-16-120. We evaluate the ranking consistency of all Zero-Shot proxies directly on three datasets, with a batch size of 64 to ensure a fair comparison. In addition, the number of batches is set to 8 with ξ=1e-8 to compute ξ-GSNR score on all three datasets. Tab.1 shows that our Zero-Shot proxy ξ-GSNR consistently outperforms other proxies across different datasets, demonstrating the robustness of ξ-GSNR to different data samples. Obviously, our method significantly improves the ranking consistency by a large margin compared to most other approaches, indicating the effectiveness of the generalization and convergence theory guarantee.

4.1.2 Results in NAS-Bench-101

NAS-Bench-101 [60] is the first large-scale Benchmark for evaluating various NAS algorithms. It contains 423K

---

<table>
<thead>
<tr>
<th>Method</th>
<th>CIFAR-10 Spearman’s ρ</th>
<th>Kendall’s τ</th>
<th>CIFAR-100 Spearman’s ρ</th>
<th>Kendall’s τ</th>
<th>ImageNet-16-120 Spearman’s ρ</th>
<th>Kendall’s τ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNIP [23]</td>
<td>0.638</td>
<td>0.472</td>
<td>0.637</td>
<td>0.474</td>
<td>0.578</td>
<td>0.433</td>
</tr>
<tr>
<td>GraSP [54]</td>
<td>0.551</td>
<td>0.385</td>
<td>0.549</td>
<td>0.388</td>
<td>0.553</td>
<td>0.395</td>
</tr>
<tr>
<td>GradNorm [1]</td>
<td>0.637</td>
<td>0.469</td>
<td>0.637</td>
<td>0.472</td>
<td>0.578</td>
<td>0.430</td>
</tr>
<tr>
<td>SynFlow [51]</td>
<td>0.777</td>
<td>0.581</td>
<td>0.763</td>
<td>0.568</td>
<td>0.751</td>
<td>0.561</td>
</tr>
<tr>
<td>TE-NAS [7]</td>
<td>0.376</td>
<td>0.257</td>
<td>0.350</td>
<td>0.239</td>
<td>0.335</td>
<td>0.228</td>
</tr>
<tr>
<td>Zen-Score [30]</td>
<td>0.251</td>
<td>0.236</td>
<td>0.260</td>
<td>0.236</td>
<td>0.319</td>
<td>0.277</td>
</tr>
<tr>
<td>NASWOT [35]</td>
<td>0.691</td>
<td>0.522</td>
<td>0.704</td>
<td>0.535</td>
<td>0.700</td>
<td>0.527</td>
</tr>
<tr>
<td>GradSign [66]</td>
<td>0.808</td>
<td>0.619</td>
<td>0.792</td>
<td>0.600</td>
<td>0.783</td>
<td>0.593</td>
</tr>
<tr>
<td>ZiCo [28]</td>
<td>0.800</td>
<td>0.610</td>
<td>0.810</td>
<td>0.610</td>
<td>0.790</td>
<td>0.600</td>
</tr>
<tr>
<td>FLOPs</td>
<td>0.733</td>
<td>0.541</td>
<td>0.708</td>
<td>0.517</td>
<td>0.673</td>
<td>0.487</td>
</tr>
<tr>
<td>Parameters</td>
<td>0.751</td>
<td>0.576</td>
<td>0.727</td>
<td>0.552</td>
<td>0.690</td>
<td>0.519</td>
</tr>
<tr>
<td>ξ-GSNR</td>
<td><strong>0.845</strong></td>
<td><strong>0.661</strong></td>
<td><strong>0.840</strong></td>
<td><strong>0.658</strong></td>
<td><strong>0.793</strong></td>
<td><strong>0.608</strong></td>
</tr>
</tbody>
</table>

Table 1. Comparison Ranking Consistency of Zero-Shot proxies in NAS-Bench-201 search space.

<table>
<thead>
<tr>
<th>Method</th>
<th>Spearman’s ρ</th>
<th>Kendall’s τ</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNIP [23]</td>
<td>0.191</td>
<td>0.131</td>
</tr>
<tr>
<td>GraSP [54]</td>
<td>0.329</td>
<td>0.223</td>
</tr>
<tr>
<td>GradNorm [1]</td>
<td>0.265</td>
<td>0.182</td>
</tr>
<tr>
<td>SynFlow [51]</td>
<td>0.360</td>
<td>0.246</td>
</tr>
<tr>
<td>TE-NAS [7]</td>
<td>0.084</td>
<td>0.056</td>
</tr>
<tr>
<td>Zen-Score [30]</td>
<td>0.261</td>
<td>0.187</td>
</tr>
<tr>
<td>NASWOT [35]</td>
<td>0.327</td>
<td>0.220</td>
</tr>
<tr>
<td>GradSign [66]</td>
<td>0.422</td>
<td>0.296</td>
</tr>
<tr>
<td>FLOPs</td>
<td>0.422</td>
<td>0.297</td>
</tr>
<tr>
<td>Parameters</td>
<td>0.423</td>
<td>0.298</td>
</tr>
<tr>
<td>ξ-GSNR</td>
<td><strong>0.615</strong></td>
<td><strong>0.434</strong></td>
</tr>
</tbody>
</table>

Table 2. Comparison Ranking Consistency of Zero-Shot proxies in NAS-Bench-101 search space on CIFAR-10.
unique architectures trained on the CIFAR-10 dataset. This benchmark provides the test accuracy of each architecture, making it possible to determine the ranking consistency between the proxy score and the corresponding accuracy by using Spearman’s $\rho$ and Kendall’s $\tau$, respectively.

Following the experimental settings of GradSign [66], we randomly sample 4500 architectures from the Benchmark to compare ranking consistency. As shown in Tab.2, our $\xi$-GSNR proxy performs better than either existing empirical or theoretical proxies, as well as two strong baselines (i.e., Parameters and FLOPs), demonstrating the superiority of our $\xi$-GSNR proxy in predicting network accuracy.

### 4.1.3 Results in NAS Design Space (NDS)

NAS Design Space (NDS) [39] summarized and analyzed various network design spaces: DARTS [32], ENAS [38], PNAS [31], NASNet [68], AmoebaNet [41]. Each network design space consists of numerous individual architectures trained on CIFAR-10 dataset. We keep the same batch size as 64 when computing the proxy score for each space. The results are presented in Tab.3. Though some proxies fail to rank specific architectures positively, our method still achieves consistently optimal ranking consistency across different design spaces, showing the advantages of $\xi$-GSNR proxy in evaluating different candidate architectures.

#### 4.2. Zero-Shot Search

To search for the optimal architecture without training, we incorporate $\xi$-GSNR into Zero-Shot search algorithm to verify the effectiveness of our proxy. Specifically, we adopt the pruning-based search algorithm [7] in the cell-based search space, including NAS-Bench-201 [15] and DARTS [32] search space, and adopt the evolution algorithm in chain-style ProxylessNAS [5] and ViT-Like BurguerFormer [59] search space. The details of the search algorithm are provided in Appendix B.

#### 4.2.1 Search results in NAS-Bench-201 space

In NAS-Bench-201, the super-network is stacked by cells, and reduction cells

<table>
<thead>
<tr>
<th>Method</th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
<th>ImageNet-16-120</th>
<th>Search Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Validation</td>
<td>Test</td>
<td>Validation</td>
<td>Test</td>
</tr>
<tr>
<td>Optimal</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENAS [38]</td>
<td>91.61</td>
<td>94.37</td>
<td>73.49</td>
<td>73.51</td>
</tr>
<tr>
<td>RSPS [29]</td>
<td>37.51±3.19</td>
<td>53.89±0.58</td>
<td>13.37±2.35</td>
<td>13.96±2.33</td>
</tr>
<tr>
<td>DARTS [32]</td>
<td>80.42±3.58</td>
<td>84.07±3.61</td>
<td>52.12±5.55</td>
<td>52.31±5.77</td>
</tr>
<tr>
<td>GDAS [14]</td>
<td>39.77±0.00</td>
<td>54.30±0.00</td>
<td>15.03±0.00</td>
<td>15.61±0.00</td>
</tr>
<tr>
<td>SETN [13]</td>
<td>89.89±0.08</td>
<td>93.61±0.09</td>
<td>71.34±0.04</td>
<td>70.70±0.30</td>
</tr>
<tr>
<td>SynFlow [51]</td>
<td>84.04±0.28</td>
<td>87.64±0.00</td>
<td>58.86±0.06</td>
<td>59.05±0.24</td>
</tr>
<tr>
<td>NASPOT [35]</td>
<td>89.83±0.75</td>
<td>93.12±0.52</td>
<td>69.89±1.87</td>
<td>69.94±1.88</td>
</tr>
<tr>
<td>GradSign [66]</td>
<td>89.55±0.89</td>
<td>92.91±0.99</td>
<td>69.35±1.70</td>
<td>69.48±1.70</td>
</tr>
<tr>
<td>TE-NAS [7]</td>
<td>89.84±0.61</td>
<td>93.31±0.47</td>
<td>70.22±1.32</td>
<td>70.33±1.28</td>
</tr>
<tr>
<td>ξ-GSNR</td>
<td>94.05±0.18</td>
<td>91.20±0.43</td>
<td>71.82±0.49</td>
<td>72.18±0.74</td>
</tr>
</tbody>
</table>

Table 4. Search results in NAS-bench-201. We report the average performance with four independent runs. “Optimal” represents the highest accuracy for each dataset. 1: We report the results with N=100 that denotes the number of networks sampled in each run.
being maintained as residual blocks. Each normal cell consists of six edges and five candidate operations associated with each edge. Similar to TE-NAS [7], we set the pruning number of operations associated with each edge to 1 for each step, and directly search for the optimal architecture on three different datasets. We conduct four independent search runs with different random seeds and report the mean and standard deviation in Tab.4. We achieve the best performance when compared with either training-based or other training-free methods. In particular, \(\xi\)-GSNR obtains higher test accuracy than TE-NAS [7], demonstrating that our proxy is superior to Neural Tangent Kernel (NTK) or linear region theory in predicting architecture performance.

\subsection*{4.2.2 Search results in DARTS space}

We leverage \(\xi\)-GSNR proxy to search for optimal architectures in DARTS search space directly on CIFAR-10 and ImageNet, respectively. To conduct the search, we adopt the algorithm proposed in TE-NAS [7]. We gradually prune low-importance operations predicted by the Zero-Shot proxy until the single-path cell is reached. The searched architecture is then evaluated following the standard experimental settings [32] to ensure a fair comparison.

Tab.5 summarizes the performance of searched architectures on CIFAR-10 and ImageNet datasets. In general, Zero-Shot NAS has a faster search speed than One-Shot NAS, indicating the efficiency of training-free search. More specifically, \(\xi\)-GSNR achieves the highest average test accuracy among all methods, demonstrating the effectiveness of our proxy in evaluating architecture performance. When searching directly on the ImageNet dataset, the performance and search speed have also been improved, benefiting from the novel Zero-Shot NAS proxy.
The ProxylessNAS \[5\] space is the popular chain-style search space. We search for the kernel size \{3, 5, 7\} and expansion ratio \{3, 6\} for the bottleneck blocks on each layer. We use the evolutionary algorithm to search for architectures with around 400M FLOPs constraints. The search iteration elapses 2000 steps with a population of 128. Finally, the optimal architecture is retrained following DNA \[25\] settings.

As shown in Tab.6, our searched architecture yielded the highest top-1 accuracy 78.2\% with only 409M FLOPs. Impressively, our search cost only requires 0.3 GPU-days, surpassing all the other training-based methods. This highlights the effectiveness and efficiency of our Zero-Shot NAS proxy in discovering excellent architectures.

4.2.4 Search results in BurgerFormer space

We conduct the experiments on ImageNet using BurgerFormer \[59\] search space, which is a ViT-like structure. We use the evolutionary algorithm to search for architectures within 30M Params and 5.0G FLOPs. The results in Tab.7 demonstrate that our method not only significantly outperforms other NAS methods, but also outperforms the manually designed networks under similar resource constraints. Even when compared to Swin-S with 50M Params, we still obtain a higher accuracy.

Moreover, we evaluate the performance of our searched architecture in BurgerFormer space for object detection and instance segmentation on COCO datasets. The pre-trained structure is employed as the backbone for the Mask R-CNN detector. The results in Tab.8 show that we achieve 45.0 AP\(b\) and 40.7 AP\(M\) on object detection and instance segmentation, clearly surpassing that of ResNet-50 and the other handcrafted ViT-like structures.

4.3. Ablation Study

4.3.1 The hyper-parameters of GSNR

There are three hyper-parameters that may fluctuate the ranking consistency when calculating \(\xi\)-GSNR proxy in Eq.(10). We randomly sample 200 architectures from NAS-Bench-201 and compute their \(\xi\)-GSNR proxy scores by varying batch sizes \(|\mathcal{B}|\), batch numbers \(N\), and random \(\xi\).

Batch Sizes \(|\mathcal{B}|\). The batch sizes \(|\mathcal{B}|\) varies from \{8, 16, 32, 64, 128, 256, 512\} while keeping other parameters constant. As shown in Fig. 3(a), a batch size of 64 produces the highest ranking consistency on most datasets. Though the value on CIFAR-10 may not reach the highest, we still keep the batch size to 64 on all experiments for a fair comparison.

Batch Numbers \(N\). The Batch numbers \(N\) varies from \{2, 4, 6, 8, 10, 12, 14\} to compute \(\xi\)-GSNR proxy. Fig.3(b) indicates that \(N=8\) is sufficient to obtain the best correlation. Hence, we adopt 8 batches in other search spaces and datasets as well, considering the trade-off between efficiency and accuracy.

Random \(\xi\). The random \(\xi\) is set to an extremely small value for stabilizing the computation. Fig.3(c) demonstrates that \(\xi=1e^{-8}\) is stable enough to improve the ranking. So we keep \(\xi=1e^{-8}\) for all experiments.

4.3.2 The variants of GSNR

Here we investigate the effect of GSNR variants, including: (1) gradient’s squared mean; (2) inverse of gradient’s variance; (3) inverse of gradient’s variance with \(\xi\); (4) vanilla GSNR; and (5) our practical \(\xi\)-GSNR. We randomly sample 200 architectures from NAS-Bench-201 and evaluate their ranking consistency on ImageNet-16-120. In Tab.9, we ob-
Figure 3. The ranking consistency between test accuracy in NAS-Bench-201 and $\xi$-GSNR computed under varying hyper-parameters, including (a) Batch Size $|B|$, (b) Batch Numbers $N$, and (c) Random $\xi$.

<table>
<thead>
<tr>
<th>Method</th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
<th>ImageNet-16</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@5%</td>
<td>P@10%</td>
<td>P@5%</td>
</tr>
<tr>
<td>SNIP</td>
<td>0.064</td>
<td>0.114</td>
<td>0.060</td>
</tr>
<tr>
<td>GradNorm</td>
<td>0.086</td>
<td>0.192</td>
<td>0.092</td>
</tr>
<tr>
<td>Zen-Score</td>
<td>0.158</td>
<td>0.379</td>
<td>0.167</td>
</tr>
<tr>
<td>NASWOT</td>
<td>0.163</td>
<td>0.383</td>
<td>0.170</td>
</tr>
<tr>
<td>$\xi$-GSNR</td>
<td>0.366</td>
<td>0.475</td>
<td>0.394</td>
</tr>
</tbody>
</table>

Table 10. Comparison the Top-K correlation in NAS-Bench-201.

Table 11. Comparison Ranking Consistency with different initializations in NAS-Bench-201 using Kendall’s $\tau$ rank.

4.3.3 The Top-K Correlation

We compute the Top-K correlation on NAS-Bench-201 across three different datasets. Specifically, $P@TopK = \frac{\#(r<KM \land n<KM)}{KM}$, which is defined in [33], representing the proportion of predicted top-K% in true top-K% architectures. The results in Tab.10 show that our method achieves the highest $P@Top5\%$ and $P@Top10\%$, indicating our ability to identify superb networks.

4.3.4 Different initializations

To demonstrate the robustness of $\xi$-GSNR proxy with different initializations, we conduct the experiments in NAS-Bench-201 on three datasets. Specifically, we use two types of initializations, including random seeds (seed=0, 1, 2) and different initialization ways (orthogonal, xavier, kaiming). The results in Tab.11 show that we achieve the stable Kendall’s $\tau$ with quite a low standard deviation when using different initializations, demonstrating the robustness of $\xi$-GSNR proxy.

5. Conclusion

Recently, Zero-Shot NAS has been attracting more and more attention due to its search efficiency. However, designing an effective proxy can be cumbersome and challenging. In this study, we theoretically analyzed that GSNR is strongly correlated with network generalization and convergence. Further, to enhance the performance and stabilize the computation, we develop $\xi$-based gradient signal-to-noise ($\xi$-GSNR) to predict architecture accuracy at initialization. Extensive experiments demonstrate the effectiveness and efficiency of the $\xi$-GSNR proxy. In the future, we will further explore more advanced Zero-Shot proxies to more accurately predict architecture performance.

Acknowledgements

The research of Yu Hu, Zihao Sun, Longxing Yang, Shun Lu, and Jilin Mei is supported by the National Natural Science Foundation of China under Grant No. 62176250 and No. 62203424. The research of Wenxiao Zhao and Yu Sun is supported by National Key Research and Development Program of China (2018YFA0703800).
References

[14] Xuanyi Dong and Yi Yang. Searching for a robust neural architecture in four gpu hours. In CVPR. 2019. 1, 2, 6, 7
[29] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In Uncertainty in artificial intelligence, 2020. 6
[31] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In ECCV. 2018. 6, 7
[34] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In ICLR. 2021. 8


[38] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In ICML, 2018. 1, 2, 6, 7


[40] Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In ICML, 2017. 2

[41] Esteban Real, Alok Aggarwal, Yaning Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In AAAI, 2019. 1, 2, 6, 7

[42] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In ICML, 2017. 1, 2


[58] Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient architecture search. In ICLR, 2019. 1, 2, 7

[59] Longxing Yang, Yu Hu, Shun Lu, Zihao Sun, Jilin Mei, Yinhe Han, and Xiaowei Li. Searching for burgerformer with micro-meso-macro space design. In ICML, 2022. 6, 8


[68] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In CVPR, 2018. 1, 2, 6, 7