Exploiting Network Compressibility and Topology in Zero-Cost NAS

Anonymous[1]

[1]Anonymous Institution

Abstract Neural Architecture Search (NAS) has been widely used to discover high-performance neural network architectures over manually designed approaches. Despite their success, current NAS approaches often require extensive evaluation of many candidate architectures in the search space, or training of large super networks. To reduce the search cost, recently proposed zero-cost proxies are utilized to efficiently predict the performance of an architecture. However, while many new proxies have been proposed in recent years, relatively little attention has been dedicated to pushing our understanding of the existing ones, with their mutual effects on each other being a particularly – but not entirely – overlooked topic. Contrary to that trend, in our work, we argue that it is worth revisiting and analysing the existing proxies in order to further push the boundaries of zero-cost NAS. Towards that goal, we propose to view the existing proxies through a common lens of network compressibility, trainability, and expressivity, as discussed in pruning literature. Notably, doing so allows us to build a better understanding of the high-level relationship between different proxies as well as refine some of them into their more informative variants. We leverage these insights to design a novel saliency and metric aggregation method informed by compressibility, orthogonality and network topology. We show that our proposed methods are simple but powerful and yield some state-of-the-art results across popular NAS benchmarks.

1 Introduction

The goal of Neural Architecture Search is to automatically identify high-performance neural network architectures with respect to some measure of success. NAS has three essential components (He et al., 2021): an appropriate search space of possible architectures; a search algorithm; and a metric that quantifies the performance of the architectures. The choice of metric determines the geometry of the search space and, thus, the trajectory of the search algorithm. So, finding appropriate metrics is of foundational importance to NAS. Metrics that rapidly estimate architecture performance are used to reduce the search time. Such metrics are often referred to as "zero-cost proxies" (White et al., 2021). There has been extensive work proposing novel metrics; however, means for improving existing metrics have received less attention. On one level, there is an implicit assumption that parameter-level information should be aggregated into a network-level score via summation (Abdelfattah et al., 2021). At a more coarse-grained perspective, while there has been an increase in research on combining zero-cost metrics (Chen et al., 2021; Krishnakumar et al., 2022; White et al., 2021), this field is still largely unexplored. In this work, we attempt to tackle both levels of composition: composing saliencies into metrics and metrics into combined proxies.

We begin, in Section 3.1, by comparing a set of existing metrics which we initially believe are ripe for combination. These metrics are divided into two groups: those that take the gradient's properties into account and those that take the activation patterns into account. Based on this, in Section 3.2, we explore what the notions of compressibility and layer-wise partitions have to offer gradient- and activation-based metrics. This is initially motivated by a reframing of the factors that contribute to the success of ZiCo (Li et al., 2023). However, by generalising the tenets that underpin SSNR and ZiCo, we draw links to the ideas behind Naswot and thus produce a useful and

 © 2023 the authors, released under CC BY 4.0

informative Naswot variant. In Section 3.3, we give motivation for and provide an example of direct metric composition. This is then applied to the metrics that were found in section 3.2. Finally, in Section 4, we consider the performance of our proposed metrics across a number of search spaces. Our main contributions can be summarised as follows:

- We reframe ZiCo and enhance existing techniques for aggregating saliency scores through the lens of network compressibility.
- We generalise the tenets of this approach to bolster Naswot, an existing proxy for expressivity.
- We demonstrate the feasibility of direct metric composition and use it, together with the above contributions, to produce a state-of-the-art zero-cost proxy.

2 Related Work

Gradient-based Saliency Scores. Pruning-at-initialisation (PaI) considers whether it is possible to reduce the size of an initialised network whilst maintaining (or even improving) performance (Wang et al., 2022). One way to reduce the number of parameters in the network is by identifying a saliency (importance) metric on the parameters and removing those with the lowest scores. Lee et al. (2019) define a parameter as salient if its removal would significantly affect the loss. Wang et al. (2020) were more interested in the training dynamics of the network, defining a parameter's saliency around its affect on the gradient norm. From these respective approaches, they produce saliency scores, SNIP and GraSP. Tanaka et al. (2020) generalized the notion of synaptic saliency used by both SNIP and GraSP. Based on this generalisation, they proposed a saliency score, Synflow, that assessed the magnitude of a parameter's contribution to the flow of synaptic strengths.

Zero-Cost NAS. Finding low-cost metrics that can predict network performance accurately is one of the major challenges in NAS. Those that use no data or only a minimal amount of data are referred to as zero-cost metrics; they allow for large search spaces to be assessed in reasonable times. Many of the most successful zero-cost metrics have been inspired by the pruning-at-initialisation literature. Abdelfattah et al. (2021) turned saliency scores into zero-cost metrics by taking their sum, transforming Synflow, SNIP, and GraSP into zero-cost metrics. These, along with other metrics such as ZiCo (Li et al., 2023) and NTK approaches (Chen et al., 2021; Shu et al., 2019), use gradient information to provide a proxy for a network's trainability. There are also many zero-cost scores that consider properties of the linear regions of the network: Naswot (Mellor et al., 2021), ZenNAS (Lin et al., 2021), etc. These are interpreted as estimating the network's expressivity.

Comparing and Combining Different Zero-Cost Metrics. Recently, researchers have begun to investigate methods for combining different zero-cost metrics altogether. Chen et al. (2021) were among the the first to consider this. They hypothesised that metrics corresponding to trainability and expressivity should be combined. Expressivity refers to the complexity of the function that the network can represent. Trainability refers to the network's ability to effectively and faithfully direct feedback signals to its parameters. They used the condition number of the NTK as a measure of trainability and the expected number of linear regions as a measure of expressivity. Finally, Chen et al. (2021) ranked the architectures with respect to these distinct metrics and averaged the ranks, resulting in a combined proxy, TE-NAS. Although TE-NAS was a useful first step, it was an indirect means of combining metrics and didn't ensure that its metrics were tracking different network properties. To alleviate problems like this, Krishnakumar et al. (2022) investigated the relatedness of different metrics. They evaluated the conditional entropy of metrics to demonstrate which ones combine best and provide the most complementary information. We utilise similar techniques to ensure that the metrics we consider in this paper track different aspects of the network.

3 Method

In the following three sections, we have the overarching goal of improving methods for aggregating parameter scores into metrics and metrics into combined proxies. We investigate a few specific metrics that turn out to have interesting properties.

3.1 Comparing Existing Metrics

We begin by investigating existing metrics, to better understand their relationship, strengths and shortcomings. In particular, our focus is on the three state-of-the-art metrics: ZiCo (Li et al., 2023), Synflow (Tanaka et al., 2020) and Naswot (Mellor et al., 2021), and the goal is to analyse them in terms of redundant and/or complementary behaviour between them. Although at first they might appear quite different from each other – ZiCo utilises gradient statistics of the training data, Synflow is data-independent measure of synaptic flow and Naswot measures diversity in the activation patterns of a network – considering their common trait of achieving strong empirical performance, we wonder if there might be a single way to think about their success. In order to do that, we begin by investigating quantitatively how much these metrics overlap in scoring different networks, if we take away the common trait of being a way to track accuracy. This is similar to what Krishnakumar et al. attempted in their work when measuring information gain from combining different metrics (Krishnakumar et al., 2022), but also focuses on a slightly different aspect. Specifically, information gain is a direct measure of how much the addition of a metric helps in predicting accuracy and, as such, was proposed as a quantitative guide to combining different metrics. Among other things, it is not commutative. On the other hand, we are interested in how different metrics relate to each other on a more fundamental level, before even considering their usefulness for NAS – by design this relationship should be commutative and disregard accuracy as a common factor behind their design. Therefore, rather than relying on information gain, we propose to utilise partial correlation (Rummel, 1976) to achieve that. Specifically, partial correlation treats the task-dependent accuracy as a covariate, measuring the agreement between the rankings produced by two metrics after disregarding the common covariate, hopefully extracting a more faithful description of how the metrics are related:

$$\rho_{XY \cdot Z} = \frac{\rho_{XY} - \rho_{XZ} \cdot \rho_{YZ}}{\sqrt{(1 - \rho_{XZ}^2)(1 - \rho_{YZ}^2)}} \tag{1}$$

where X, Y represent metrics; Z represents the accuracy; and each correlation is a Kendall-Tau correlation between the architecture rankings. These partial correlations between the three metrics are illustrated by Figure 1, left. We can observe that ZiCo correlates quite well with Synflow, which can be explained by the fact that both rely on gradient information. On the other hand, correlation between Naswot and Synflow is significantly lower, which is aligned with the fact that Naswot is based on activation patterns rather than gradient. Overall, our results seem to support the existing distinction of "expersivity" (activation-based Naswot) and "trainability" (gradient-based ZiCo and Synlow) metrics discussed in previous works (Chen et al., 2021). However, the somewhat high correlation between ZiCo and Naswot is a bit surprising and not easily explainable with the existing literature. This further motivates us to more closely investigate differences and unify concepts within and between the gradient- and activation-based classes through the lens of network compression and topology. We present this in the following sections.

3.2 Incorporating Network Compressibility and Topology into Zero-Cost NAS

In this section, we start by looking at gradient-space compressibility. This leverages the gradient to determine whether a network effectively uses the parameters and information it has available. A new interpretation of ZiCo as a measure of gradient compressibility served as the inspiration for this viewpoint. By expanding on this concept, we derive an orthogonal form of gradient compressibility,

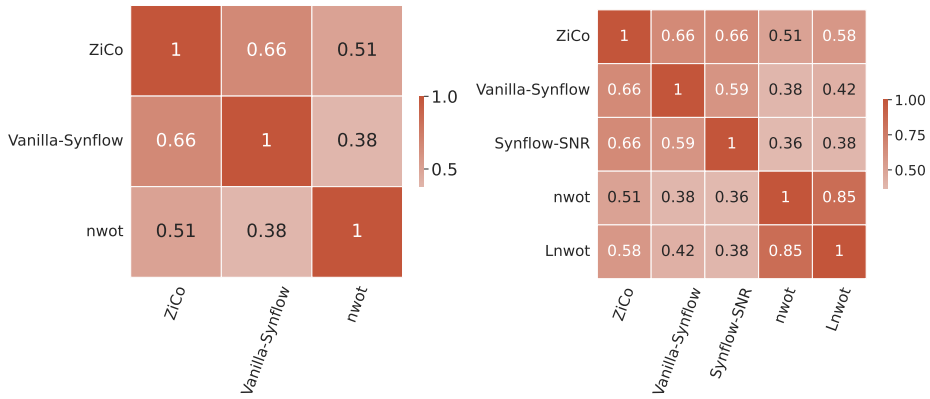

Figure 1: The partial correlation between the different metrics on NasBench201. The left image (1.a) considers a small subset of existing metrics partial correlations. The right image (1.b) considers a larger set of metrics, including some of those produced in Section 3.2.

SSNR, that can be data-independent (Synflow-SNR) and yet correlates well with ZiCo. The general tenets of compressibility and layer-wise partitions that underlie ZiCo and SSNR are used to reframe and modify Naswot into a more informative variant.

Gradient-space Compressibility. Let us consider a neural network parametrised with θ, and a minibatch of N inputs $X = \{x_i\}_{i=1}^{N}$. Function $f : \mathbb{R}^{NxI} \times \mathbb{R}^{M} \longrightarrow \mathbb{R}^{N}$ is a composition of the network and a loss function, where the loss is applied element-wise (unreduced) to the outputs of the neural network for different elements of the input minibatch X, M is the number of weights $M = |\theta|$ and I is dimensionality of a single input. Training dynamics of the underlying neural network are captured by the related Jacobian matrix:

$$J_X = \begin{bmatrix} \nabla_\theta f^1 \\ \vdots \\ \nabla_\theta f^N \end{bmatrix} = \begin{bmatrix} \frac{\partial f^1}{\partial \omega_1} & \cdots & \frac{\partial f^1}{\partial \omega_M} \\ \vdots & \ddots & \\ \frac{\partial f^N}{\partial \omega_1} & & \frac{\partial f^N}{\partial \omega_M} \end{bmatrix} \tag{2}$$

where f^i is the i-th output of the function (loss of the i-th element in X). We first remark that both ZiCo and Synflow (and in fact a broader class of proxies based on gradient-based scores, as explained by Tanaka et al. (2020)) are both functions of this Jacobian matrix, although they differ in what operations are applied to it, as well as what input and/or loss function are used. Regardless of these differences, the common goal is to reduce the information contained in this matrix to a scalar value representative of a network's performance.

ZiCo in particular argues about the relationship between column-wise means and variance and a network's generalizability and convergence rate, motivating their design. Despite achieving strong empirical results, their interpretation and link with theoretical analysis have certain shortcomings: 1) it suggests that the metric is related to the convergence speed, which is beneficial for predicting accuracy; however, we know there exist networks that converge slower to better results – ZiCo can correctly rank them despite the backing theory suggesting that those converging faster should be preferred. 2) somewhat related, the theoretical analysis is performed only for simple networks and it is unclear how applicable it is to networks of practical size. Because of these shortcoming, we propose a different interpretation of ZiCo's strong empirical performance. Specifically, we would argue that ZiCo effectively measures the compressibility of matrix J along the columns. It fixes a parameter and calculates the expectation (μ) and variance (σ^2) of the gradient's size over the

samples. By aggregating μ/σ for each parameter, it assesses the compressibility of the gradient. The gradient is compressible in relation to the samples if the variance is large, as the training dynamics are likely dominated by a small subset of the inputs, specifically those that produce the largest gradient size. Similarly, a smaller variance implies that the gradient's size for each parameter is invariant over the inputs, thus ensuring the network effectively uses all the data it has available. This insight may explain the success of ZiCo as a test for whether the training dynamics use the data-set as a whole. Importantly, compression-based interpretation – in its generalized form – is independent from either training data or performing column-wise operations, suggesting that it can be beneficial in a broader range of cases.

Motivated by this realisation, we consider whether an orthogonal measure of compressibility is appropriate. Once again, considering J_X, we ask whether the compressibility of this matrix along the columns is informative. Intuitively, this is measuring how important each parameter is; hence, we are inspired to consider whether the pruning literature is relevant and choose to replace the gradient, $\nabla_\theta f$, with an analogous concept: a saliency score. In the pruning-at-initialization literature, the saliency, $S(\theta_i)$, of a parameter, θ_i, can be viewed as a signal that measures how much the parameter contributes to the final model accuracy. If two architectures have similar saliency strengths, the one whose parameters have more diverse saliency signals should have a better chance of being pruned without harming accuracy, i.e., higher compressibility. From this, we produce a metric (SSNR) which effectively, ignoring the technical differences outlined below, aggregates μ/σ, where the expectation (μ) and variance (σ^2) of the saliency is taken over the parameters. This has a natural interpretation in the NAS-context where the score for a given model is calculated by summing over the signals produced by all its parameters:

$$S_n = \sum_{i}^{N} S(\theta_i). \tag{3}$$

During pruning, the parameters with lower saliency scores will be dropped, but the model accuracy is unlikely to be strongly affected. This likely means that there exist other models that have a lower overall saliency (S_n), but with a similar ground truth accuracy. So, we may get architectures with similar accuracy but potentially different zero-cost scores. This illustrates a potential mismatch between zero-cost scores and model accuracy. The signal-to-noise ratio of saliency (SSNR) helps to alleviate this problem. Before we explicitly define the final form of SSNR, it is worth considering that ZiCo also utilises a layer-wise partition of the edges; is such a partition suitable here?

In a network, every path passes through one and only one parameter in a given layer. The consequences of this are more formally addressed in Theorem 2 of Tanaka et al. (2020) which states that each layer of the network receives the same total synaptic saliency. Wider layers will distribute this saliency over more parameters, reducing the noise. This justifies the use of a layer-wise partition as the noise (standard deviation) we wish to extract is within layers, not between them. Defining $\theta_i \in \Theta^l$ if and only if θ_i is a parameter in layer l then the signal and noise are given by:

$$S_n^l = \sum_{\theta_i \in \Theta^l} S(\theta_i) \quad \text{and} \quad \sigma^l = \sqrt{\frac{1}{|\Theta^l|} \sum_{\theta_i \in \Theta^l} \left(S(\theta_i) - \frac{S_n^l}{|\Theta^l|} \right)} \tag{4}$$

In equation (1), we see that in the NAS context the sum of saliency scores (S_n) is more appropriate than the expectation (μ). Therefore, the signal-to-noise ratio's are aggregated to give:

$$SSNR = \sum_{l} log \left(\frac{S_n^l}{\sigma^l} \right) \tag{5}$$

The logarithm is used to stabilise the computation, ensuring no individual layers dominate the score. Sometimes we will distinguish the measure of saliency that SSNR acts on, and we will refer to these SSNR instances as Synflow-SNR, SNIP-SNR, and GraSP-SNR. In summary, ZiCo measures the compressibility of the gradient across the rows, ensuring that all inputs are used effectively in the training dynamics. Similarly, SSNR measures compressibility of synaptic saliency across the rows, ensuring that all parameters are used effectively in the training dynamics.

Activation-space compressibility. We have showed that the success of ZiCo and SSNR can be attributed to the fact that both approaches consider similar concepts of gradient spread (which we call compressibility) with a form of layer-wise granularity. Considering our pilot study presented in Section 3.1 which showed that ZiCo and Naswot share similarities in their scoring behaviour, we now ask the question if a similar notion of compressibility could have an interpretation with respect to activation patterns? To answer this question, we need to realise the activation patterns are maximally compressed when they are as orthogonal as possible, as this ensures that the linear regions are distributed over the inputs and thus are effectively utilising all available neurons. Mellor et al. (2021) use the determinant to assess whether the activation patterns are compressed over the input space (decorrelated over a mini-batch), and as such, whether all neurons provide novel information. This is motivated by the assumption that inputs with decorrelated activation patterns will be easier for the network to separate. Overall, we would argue that, by design, Naswot is in fact a direct measure of compressability in the activation space. However, Naswot doesn't consider a key part of what bolstered ZiCo and SSNR: a layer-wise partition. Might such a partition be sensible also in its case?

For a network with L layers, we consider L Naswot kernels defined by the layer-wise activation patterns. The diagonal terms of the Naswot kernel in each layer are the number of ReLU neurons in that layer. Theorem 1 of Hanin and Rolnick (2019) implies that the number of neurons in a layer scales like the number of "practically expressible" linear regions that the layer contributes to the network. Furthermore, on a more practical basis, network-level Naswot kernels are untenable if batches are reshaped or subdivided between layers (Duan et al., 2021). Hence, Naswot clearly has a natural and useful interpretation under a layer-wise decomposition. Layer-wise Naswot, Lnwot, assigns each layer a score of $log|K^l|$ and sums them into a final metric, where the log's on each layer help to ensure that no layer dominates the final score.

3.3 Directly Combining Compressibility Metrics

In the previous two sections we have explored improvements to existing metrics. In this section, we investigate means for combining a gradient-based metric (i.e., SSNR) with a activation-based approach (Naswot). But we do this through a more general goal of interest: how to faithfully combine any two chosen metrics. These combinations may be direct, acting on network-, layer-, or even parameter- level scores, or indirect, acting on architecture rankings. As an approach to metric combination, Chen et al. (2021) averaged the ranks of the architectures with respect to the constituent metrics, an approach that we will abbreviate as the "rank average" or "RankAve". RankAve has the advantage that it equally weights the contribution of each metric, this is empirically demonstrated in row 1 of Table 1. However, it also has a number of drawbacks. Combining ranks is feasible for some search algorithms and search spaces, however is intractable in many examples (Lin et al., 2021). Furthermore, it is indirect and arguably doesn't create a combined metric, instead offering a method to combine metrics. These insights inspire a general goal of discovering more direct and flexible means to combine metric scores. In this section, we consider a first instance of this more general goal before applying it to distinct metrics of interest, SSNR and Lnwot.

Direct Composition.
In order to assess potential forms of a combined metric, let's define two arbitrary metrics, τ_i and τ_j.

We will assume these metrics are independent random variables, in which samples of the variable come from evaluating it on a network. We define, for $k \in \{i, j\}$, $\mu_k = \mathbb{E}[\tau_k]$ and $\sigma_k^2 = Var(\tau_k)$. What operation can we apply to compose these two metrics such that neither dominates the variance? The most natural binary operation is addition. Clearly, $Var(\tau_i + \tau_j) = \sigma_i^2 + \sigma_j^2$. But what is the effect of the variance on the rankings? Suppose that $\sigma_i \gg \sigma_j$, then (Alsmeyer, 2011):

$$P(|(\tau_i + \tau_j) - (\mu_i + \mu_j)| \geq k) \leq \frac{\sigma_i^2 + \sigma_j^2}{k} = \mathcal{O}(\sigma_i^2) \tag{6}$$

This indicates that the distributional properties of $\tau_i + \tau_j$ are dominated by τ_i and hence the overall ranking of the architectures is dominated by τ_i. As the variances of the metrics are unlikely to be similar, the metric with a larger variance will dominate. After ruling out addition, the next simplest operation on the reals is multiplication (Goodman, 1962):

$$Var(\tau_i \cdot \tau_j) = \sigma_i^2\sigma_j^2 + \mu_j^2\sigma_i^2 + \mu_j^2\sigma_i^2 = \sigma_i^2\sigma_j^2 \left[1 + (\mu_j/\sigma_j)^2 + (\mu_i/\sigma_i)^2\right] \tag{7}$$

This indicates a more nuanced relationship between the metrics determines the rankings. Although certainly not guaranteed, if the metrics' μ_k and σ_k scale similarly and have similar distributional characteristics, then this method prevents either metric from dominating the variance. One concern is that, even when using the metrics with the lowest correlation, it is not safe to assume independence. In spite of the size of these assumptions, we find evidence that, for Lnwot and SSNR, the contributions of the distinct metrics to the combined scores are well-balanced (Table 1). Although there are presumably superior operations to multiplication for direct composition, this approach to directly combining metrics simply aims to serve as a first attempt at doing so. We leave it to future work to further investigate methods for balancing metrics' variances in direct metric composition, potentially by considering $\left(\tau_i^\alpha \cdot \tau_j\right)$, for a suitable choice of α.

A final point to note is that by simply multiplying the final Lnwot and SSNR scores, much of the layer-wise information that is collected by these metrics is lost. As such, we consider a layer-wise multiplication of the scores and then sum, effectively taking the dot product of the layer-wise values of Lnwot and SSNR. This layer-wise composition allows individual layers to be assessed based on their contribution to the network and incorporates the orthogonality of Lnwot and SSNR over the layers. Bringing all of this together into a single metric, we propose a *Topological*, layer-by-layer, proxy (T-CET) which balances metrics for *Compressible Expressivity* and *Trainability*:

$$\text{T-CET} = \sum_l log\left(\frac{S_n^l}{\sigma^l}\right) \cdot log|K^l| \tag{8}$$

Table 1: The columns indicate which variant of SSNR is referred to in the row titles, and used in the combined metrics, RankAve and T-CET. The first two rows show the partial correlations of RankAve with SSNR and Lnowt. The final two rows show the partial correlations of the T-CET with SSNR and Lnowt. This demonstrates that RankAve equally weights metric contributions. However T-CET, not directly trying to so, also roughly balances the contributions. This data is derived from NasBench-TSS on Cifar-10, Cifar-100 and ImageNet-120.

	Synflow -SNR (C-10)	SNIP -SNR (C-10)	Synflow -SNR (C-100)	SNIP -SNR (C-100)	Synflow -SNR (IN)	SNIP -SNR (IN)
ParCorr(SSNR, RankAve)	0.86	0.87	0.87	0.87	0.85	0.68
ParCorr(Lnwot, RankAve)	0.86	0.85	0.86	0.85	0.85	0.85
ParCorr(SSNR, T-CET)	0.91	0.93	0.90	0.92	0.88	0.90
ParCorr(Lnwot, T-CET)	0.78	0.78	0.78	0.78	0.78	0.78

Table 2: Comparison in NATS-Bench with existing proxies for full rank-correlation analysis.

	NATSBench-TSS(NASBench201)						NTSBench-SSS					
Methods	CIFAR10		CIFAR100		ImageNet16-120		CIFAR10		CIFAR100		ImageNet16-120	
	KT	SPR	KT	SPR	KT	SPR	KT	SPR	KT	SPR	KT	SPR
Baseline												
FLOPs	0.54	0.73	0.51	0.71	0.49	0.67	0.19	0.28	0.21	0.30	0.38	0.53
#Params	0.57	0.75	0.55	0.73	0.52	0.69	0.53	0.72	0.55	0.75	*0.70*	*0.88*
Activation-Pattern Based												
Zen-score	0.29	0.38	0.28	0.36	0.29	0.40	0.50	0.69	0.52	0.71	0.69	0.87
NASWOT	0.58	0.77	*0.62*	0.80	*0.60*	0.78	0.45	0.63	0.43	0.59	0.42	0.59
Layerwise-NASWOT	0.57	0.76	0.57	0.77	0.56	0.75	0.61	0.80	0.31	0.45	0.49	0.69
Gradient-Pattern Based												
Grad_norm	0.46	0.63	0.47	0.63	0.43	0.58	0.35	0.51	0.34	0.49	0.49	0.67
Grasp	0.37	0.54	0.36	0.51	0.40	0.56	-0.09	-0.13	0.01	0.01	0.29	0.42
SNIP	0.46	0.63	0.46	0.63	0.43	0.58	0.42	0.59	0.46	0.62	0.57	0.76
Synflow	0.54	0.73	0.57	0.76	0.56	0.75	0.61	*0.81*	**0.60**	**0.80**	0.39	0.57
Zico	*0.61*	*0.80*	0.61	0.81	*0.60*	0.75	0.54	0.73	0.55	0.75	*0.70*	*0.88*
Synflow-SNR	0.60	0.79	0.60	0.79	0.59	*0.79*	*0.75*	**0.92**	*0.56*	*0.76*	**0.76**	**0.92**
SNIP-SNR	**0.63**	**0.82**	**0.63**	**0.83**	**0.62**	**0.81**	**0.76**	**0.92**	*0.56*	*0.76*	**0.76**	**0.92**
Topology Combined Score												
T-CET(synflow)	**0.63**	**0.82**	*0.62*	0.81	**0.62**	**0.81**	*0.75*	**0.92**	0.44	0.63	0.66	0.85
T-CET(SNIP)	**0.63**	**0.82**	**0.63**	*0.82*	**0.62**	**0.81**	*0.75*	**0.92**	0.45	0.63	0.66	0.85

4 Evaluation

We conduct the following types of experiments: (1) Empirical analysis of the proposed metrics on multiple NAS benchmarks with rank correlation. (2) To demonstrate that our metrics are robust for a variety of different tasks and search spaces, use a random search approach with several different zero-cost metrics. (3) We show our T-CET outperform many existing zero-cost metrics on the ZenNAS EfficientNet search space with Aging Evolution search on the CIFAR task.

4.1 Empirical Analysis on Benchmarks

This section compares our proposed metrics, including T-CET, with other zero-cost proxies on various image-classification benchmarks: NATS-Bench-TSS(NASBench201) (Dong and Yang, 2020; Dong et al., 2021), NATS-Bench-SSS, NASBench-1shot1 (Zela et al., 2020), and NASBench-Macro (Su et al., 2021). Specifically, in NATS-Bench-TSS and NATS-Bench-SSS, we proved to be highly competitive with the best existing zero-cost proxies like Zen-Score (Lin et al., 2021), NASWOT (Mellor et al., 2021), Grad_norm, Grasp, SNIP, Synflow (Abdelfattah et al., 2021), Zico (Li et al., 2023). As shown in 2, the SNR boosted significantly on gradient-based metrics like Synflow and SNIP, significantly outperforming previous SOTA methods, such as Zico, on every dataset considered. Moreover, our topological combination approach, mixing Layerwise-NASWOT with SNR, improved Synflow-SNR on every dataset for the highly topological Search Space, NATS-Bench-TSS.

To further verify our observations on NATS-Bench-TSS and NATS-Bench-SSS, we extend our experiments to NASBench-1shot1 and NASBench-Macro. As opposed to NATS-Bench-TSS and NATS-Bench-SSS, whose design was based on a fixed DARTS-like graph, NASBench-1shot1 defined three subsets from the original NASBench101. In this search space, we search to determine both the operations on the node and how their edges are connected to the parent nodes, which brings an extra dynamic to the path complexity as opposed to NATS-Bench-TSS. We observed that SNIP-SNR achieved the best performance on many different metrics. Moreover, similar to our observation on NATS-Bench-TSS, the T-CET(Synflow) approach consistently benefits from combining Layerwise-NASWOT in a topology-aware manner.

On the other hand, we observed that NASWOT achieved the best performance on NASBench-Macro. While the T-CET approaches for both Synflow and SNIP are improved from their SNR baseline.

4.2 NASlib results

To verify that our approaches are robust on different tasks and design spaces, we evaluate our methods on NASlibRuchte et al. (2020) by randomly sampling architectures from diverse search

Methods	NASBench-1shot				NasBench-Macro	
	SearchSpace1		SearchSpace2			
	KT	SPR	KT	SPR	KT	SPR
Activation-Pattern Based						
NASWOT	0.44	0.60	0.34	0.49	**0.63**	**0.83**
Layerwise-NASWOT	0.52	0.70	0.57	0.75	0.59	0.78
Gradient-Pattern Based						
Synflow	0.50	0.69	0.38	0.55	0.52	0.72
SNIP	0.43	0.59	0.47	0.64	0.5	0.69
Synflow-SNR	0.50	0.68	0.57	0.75	0.52	0.72
SNIP-SNR	**0.58**	**0.77**	**0.62**	**0.82**	0.52	0.72
Topology Combination						
T-CET(Synflow)	0.54	0.72	0.60	0.78	0.55	0.75
T-CET(SNIP)	0.55	0.74	0.60	0.79	0.55	0.75

Methods	NASBench301	TranBench101-Micro			Avg-Kt
		class_scene	class_object	jigsaw	
Baseline					
Flops	0.37	0.45	0.32	0.29	0.36
Params	0.41	0.44	0.32	0.29	0.36
Activation-Pattern Based					
Zen-score	0.37	0.50	**0.42**	0.37	**0.41**
Layerwise-NASWOT	0.41	0.39	0.31	0.32	0.36
Gradient-Pattern Based					
Zico	0.37	0.45	0.32	0.29	0.36
Synflow	0.13	**0.54**	0.42	0.33	0.36
SNIP	0.03	0.48	**0.49**	**0.38**	0.35
Synflow-SNR	0.40	0.47	0.37	0.31	0.37
SNIP-SNR	0.40	0.47	0.37	0.32	0.39
Topology Combination					
T-CET(Synflow)	**0.42**	0.48	0.38	**0.38**	**0.41**
T-CET(SNIP)	**0.42**	0.48	0.38	0.37	**0.41**

Table 3: Further Experiments on Extra NAS Benchmarks(left); Random Sample 500 architectures from each NASlib tasks and calculate separate and average Kendall-tau correlation(Right)

Table 4: Top-1 accuracies on CIFAR-10 for zero-cost proxies proposed model on ZenNAS Search-Space. Budget: model size N< 1M. 'Random':average accuracy ± std for a random search.

Method	Random	FLOPs	Grad-norm	Synflow	NASWOT	TE-Score	Zen-Score	T-CET(Synflow)	T-CET(SNIP)
CIFAR-10 (Acc.%)	93.5+-0.7	93.1	92.8	95.1	96.0	96.1	96.1	96.6	**97.2**

spaces and tasks. Specifically, we randomly sampled 500 architectures from each task and calculated the zero-cost proxy with the architectures. Then, we measure the Kendall-tau rank correlation with its ground truth performance. Afterwards, we averaged rank correlation across tasks to compare the best metrics that the ranked model performs well across different tasks. As shown in 3, both T-CET(Synflow) and T-CET(SNIP) has the best average Kendall-tau correlation across different task and datasets and shows consistent improvement from their SNR approaches, showing the effectiveness of our combination approach.

4.3 Search on Practical Search Space

Moving beyond the more theoretically driven search spaces we considered a practical example. To align with previous works, we used a search space consisting of residual blocks and bottleneck blocks as defined in ResNet, which is aligned with ZenNAS (Lin et al., 2021). As shown in 4 the model found by our T-CET(Synflow) and T-CET(SNIP) outperformed the previous best ZenNet significantly where 96.2 vs (96.6 and 97.22).

5 Conclusion

One of the main contributions of our work has been to investigate how to convert parameter-level scores into reliable and precise zero-cost proxies. We have extended the existing proxy methods and analyzed the fundamental principles of these enhancements. Additionally, we have developed a novel zero-cost proxy named T-CET by combining metric scores directly. Finally, our metrics were able to deliver state-of-the-art performance across a diverse set of NAS benchmarks such as NATSBench-TSS, NATSBench-SSS, NASBench301 and TranBench101-Micro.

6 Broader Impact Statement

After careful reflection, the authors have determined that this work presents no notable negative impacts to society or the environment.

7 Submission Checklist

1. For all authors...

   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] In the introduction we refer to the exact sections which deals with each point that we consider.

   (b) Did you describe the limitations of your work? [Yes] We particularly considered this at the end of Section 3.3

   (c) Did you discuss any potential negative societal impacts of your work? [Yes] After careful reflection, the authors have determined that this work presents no notable negative impacts to society or the environment.

   (d) Have you read the ethics author's and review guidelines and ensured that your paper conforms to them? `https://automl.cc/ethics-accessibility/` [Yes] We have considered these guidelines and, for example, as such have used numbers as well as colours when plotting heatmaps to ensure accessibility

   (a) Did you state the full set of assumptions of all theoretical results? [N/A] No theoretical results are presented.

   (b) Did you include complete proofs of all theoretical results? [N/A] No theoretical results are presented.

2. If you ran experiments...

   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results, including all requirements (e.g., `requirements.txt` with explicit version), an instructive `README` with installation, and execution commands (either in the supplemental material or as a URL)? [Yes] Yes, all code will be available on github

   (b) Did you include the raw results of running the given instructions on the given code and data? [Yes] Yes

   (c) Did you include scripts and commands that can be used to generate the figures and tables in your paper based on the raw results of the code, data, and instructions given? [Yes]

   (d) Did you ensure sufficient code quality such that your code can be safely executed and the code is properly documented? [Yes]

   (e) Did you specify all the training details (e.g., data splits, pre-processing, search spaces, fixed hyperparameter settings, and how they were chosen)? [Yes]

   (f) Did you ensure that you compared different methods (including your own) exactly on the same benchmarks, including the same datasets, search space, code for training and hyperparameters for that code? [Yes] We clearly specified that NAS Benchmarks and datasets that were used in the figures and the evalutation section.

   (g) Did you run ablation studies to assess the impact of different components of your approach? [Yes]

   (h) Did you use the same evaluation protocol for the methods being compared? [Yes] The same methods were run indiscriminate of the metric.

   (i) Did you compare performance over time? [No] Our focus is on improving zero-cost metrics, which on their own are not searching methods. Therefore performance over time has secondary meaning, as it is dependant on both quality of a metric (which we assess extensively) and the searching algorithm in use.

(j) Did you perform multiple runs of your experiments and report random seeds? [No] We measure empirical correlation

(k) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We measure empirical correlation

(l) Did you use tabular or surrogate benchmarks for in-depth evaluations? [Yes]

(m) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] Will specific in Appendix

(n) Did you report how you tuned hyperparameters, and what time and resources this required (if they were not automatically tuned by your AutoML method, e.g. in a NAS approach; and also hyperparameters of your own method)? [N/A] Our results use exclusively existing training procedures/protocols to ensure fair comparison.

3. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets…

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [No] Assets are well-known in the literature and their licences can be found accordingly.

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A] No new assets are being curated/released in this work.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A] All data came from standard-practice datasets.

4. If you used crowdsourcing or conducted research with human subjects…

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not conduct research with human subjects.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] We did not conduct research with human subjects.

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] We did not conduct research with human subjects.

References

Abdelfattah, M. S., Mehrotra, A., Dudziak, Ł., and Lane, N. D. (2021). Zero-cost proxies for lightweight NAS. In *International Conference on Learning Representations (ICLR)*.

Alsmeyer, G. (2011). Chebyshev's inequality.

Chen, W., Gong, X., and Wang, Z. (2021). Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. *In International Conference on Learning Representations (ICLR)*.

Dong, X., Liu, L., Musial, K., and Gabrys, B. (2021). Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE transactions on pattern analysis and machine intelligence*, 44.

Dong, X. and Yang, Y. (2020). NAS-Bench-201: Extending the Scope of Reproducible Neural Architecture Search. In *International Conference on Learning Representations (ICLR)*.

Duan, Y., Chen, X., Xu, H., Chen, Z., Liang, X., Zhang, T., and Li, Z. (2021). Transnas-bench-101: Improving transferability and generalizability of cross-task neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Goodman, L. A. (1962). The variance of the product of k random variables. *Journal of the American Statistical Association*, 57.

Hanin, B. and Rolnick, D. (2019). Complexity of linear regions in deep networks. In *International Conference on Machine Learning (ICML)*.

He, X., Zhao, K., and Chu, X. (2021). Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212.

Krishnakumar, A., White, C., Zela, A., Tu, R., Safari, M., and Hutter, F. (2022). Nas-bench-suite-zero: Accelerating research on zero cost proxies. *In Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*.

Lee, N., Ajanthan, T., and Torr, P. H. (2019). Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations (ICLR)*.

Li, G., Yang, Y., Bhardwaj, K., and Marculescu, R. (2023). Zico: Zero-shot nas via inverse coefficient of variation on gradients. *arXiv preprint arXiv:2301.11300*.

Lin, M., Wang, P., Sun, Z., Chen, H., Sun, X., Qian, Q., Li, H., and Jin, R. (2021). Zen-nas: A zero-shot nas for high-performance deep image recognition. In *IEEE/CVF International Conference on Computer Vision, ICCV*.

Mellor, J., Turner, J., Storkey, A., and Crowley, E. J. (2021). Neural architecture search without training. In *International Conference on Machine Learning (ICML)*.

Ruchte, M., Zela, A., Siems, J., Grabocka, J., and Hutter, F. (2020). Naslib: A modular and flexible neural architecture search library. `https://github.com/automl/NASLib`.

Rummel, R. J. (1976). Understanding correlation. *Honolulu: Department of Political Science, University of Hawaii*.

Shu, Y., Cai, S., Dai, Z., Ooi, B. C., and Low, B. K. H. (2019). Nasi: Label- and data-agnostic neural architecture search at initialization. *International Conference on Learning Representations (ICLR)*.

Su, X., Huang, T., Li, Y., You, S., Wang, F., Qian, C., Zhang, C., and Xu, C. (2021). Prioritized architecture sampling with monto-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Tanaka, H., Kunin, D., Yamins, D. L. K., and Ganguli, S. (2020). Pruning neural networks without any data by iteratively conserving synaptic flow. In *Neural Information Processing Systems (NeurIPS)*.

Wang, C., Zhang, G., and Grosse, R. (2020). Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations (ICLR)*.

Wang, H., Qin, C., Bai, Y., Zhang, Y., and Fu, Y. (2022). Recent advances on neural network pruning at initialization. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

White, C., Zela, A., Ru, R., Liu, Y., and Hutter, F. (2021). How powerful are performance predictors in neural architecture search? *Advances in Neural Information Processing Systems*, 34.

Zela, A., Siems, J., and Hutter, F. (2020). Nas-bench-1shot1: Benchmarking and dissecting one-shot neural architecture search. *arXiv preprint arXiv:2001.10422*.