Training-free Neural Architecture Search through Variance of Knowledge of Deep Network Weights

Ondrej Tybl Lukas Neumann
CMP Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague
ondrej.tybl@cvut.cz lukas.neumann@cvut.cz

Abstract

Deep learning has revolutionized computer vision, but it achieved its tremendous success using deep network architectures which are mostly hand-crafted and therefore likely suboptimal. Neural Architecture Search (NAS) aims to bridge this gap by following a well-defined optimization paradigm which systematically looks for the best architecture, given objective criterion such as maximal classification accuracy. The main limitation of NAS is however its astronomical computational cost, as it typically requires training each candidate network architecture from scratch.

In this paper, we aim to alleviate this limitation by proposing a novel training-free proxy for image classification accuracy based on Fisher Information. The proposed proxy has a strong theoretical background in statistics and it allows estimating expected image classification accuracy of a given deep network without training the network, thus significantly reducing computational cost of standard NAS algorithms.

Our training-free proxy achieves state-of-the-art results on three public datasets and in two search spaces, both when evaluated using previously proposed metrics, as well as using a new metric that we propose which we demonstrate is more informative for practical NAS applications. The source code is publicly available at `https://www.github.com/ondratybl/VKDNW`.

1. Introduction

In most instances, neural network architectures are designed by authors following the field's "best-practices" or their experience, without any formal and repeatable procedure. This is however inconvenient especially in applications on a large scale. Neural Architecture Search (NAS) aims to bridge this gap by following a well-defined optimization paradigm which systematically looks for the best architecture, given objective criterion such as maximal accuracy.

The main limitation of Neural Architecture Search (NAS) is however the computational cost, because in the most basic NAS setup, it is required to train thousands or more of different deep network architectures from scratch in order just to calculate a single scalar – the objective function value, such as the classification accuracy. This severely limits practical applications of NAS as the size of feasible architecture search space is only a small fraction of the overall space of all networks.

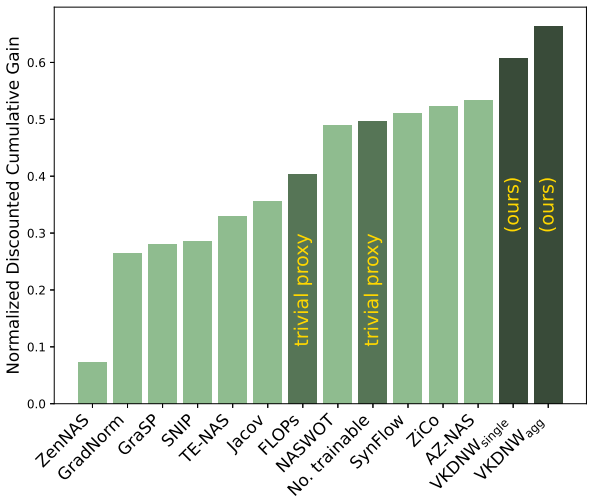

Figure 1. Training-free NAS methods on ImageNet16-120 [10]. Methods are compared by Normalized Discounted Cumulative Gain (see Sec. 3), our method (VKDNW) is the best also measured by Kendall's τ and Spearman's ρ correlations (see Table 1). Also note that simple *number of trainable layers* (below denoted $\aleph$) is significantly better trivial proxy than the number of FLOPs.

Training-free Neural Architecture Search (TF-NAS) aims to alleviate this limitation by introducing an *objective function proxy* which – unlike the actual objective function – does not require training the network. As a result, a good proxy allows the TF-NAS algorithm to explore significantly bigger portion of the network architecture search space compared to traditional NAS, and to find the best network architecture without training a single network. The crucial question is however finding an appropriate objective function proxy.

In this paper, we present a novel, principled *objective function proxy* called *Variance of Knowledge of Deep Network Weights (VKDNW)* for image classification accuracy, which allows us to **find optimal network architectures for image classification without training** them first. Our method is, to the best of our knowledge, the first successful application of Fisher Information theory [28] in the context of large deep neural networks, and as such allows us to formally describe and quantify the difficulty of network parameters' estimation process. In other words, *given a network architecture, our method estimates how easy or hard it will be to train the network.*

Additionally, we also observe that the evaluation metrics used in the TF-NAS community are not well-suited to the problem at hand, because it unnecessarily penalizes for bad proxy accuracy for networks which are not interesting, and vice-versa it does not sufficiently reward proxies which are able to accurately pick out good network architectures. Following this observation, we propose that the Normalized Discounted Cumulative Gain should be used in companion with other TF-NAS metrics, and show indeed that there are significant differences amongst previously proposed TF-NAS methods when the new metric is considered.

To summarize, we make the following contributions:

1. We introduce a novel algorithm for estimation of Fisher Information Matrix spectrum, which is tractable even for models with large number of parameters such as deep networks, that overcomes the usual problems of numerical stability.
2. We introduce a novel principled VKDNW proxy for image classification accuracy. The proxy is based on strong theoretical background and captures uncertainty in weight estimation process. It brings information that is orthogonal to the model size which then allows for efficient combination with previously proposed proxies, leading to state-of-the-art results.
3. We propose a new evaluation metric for TF-NAS proxies which is more relevant to the actual NAS objective as it concentrates on ability of given proxy to identify good networks.

2. Method

Our zero-shot proxy for image classification accuracy builds on Fisher Information theory [28], therefore we begin by a thorough analysis of Fisher Information Matrix (FIM) of the network weights estimation problem (see Sec. 2.1). We proceed by discussing challenges involved in practical application of FIM in context of large over-parametrized models that lead to only limited success in previous works and present our contributions to overcome these limitations (see Sec. 2.2). Finally. we propose a novel FIM-based proxy for NAS algorithms (Sec. 2.3).

2.1. Fisher Information

The problem of finding the optimal weights of a neural network f for the task of C-class image classification can be seen as a maximum likelihood estimation with a statistical model

$$\sigma_\theta(c \,|\, x) = \frac{\exp\left(\Psi_c(x,\theta)\right)}{\sum_{d=1}^{C} \exp\left(\Psi_d(x,\theta)\right)}, \quad c = 1, \ldots, C \quad (1)$$

where $\sigma_\theta(\cdot \,|\, x)$ denotes the a posteriori distribution of the labels given input image x and $\Psi(x,\theta) \in \mathbb{R}^C = (\Psi_1(x,\theta), \ldots, \Psi_C(x,\theta))$ is the network output (logits) given the weight vector $\theta \in \mathbb{R}^p$, i.e. the network weights. We describe the process of training as finding the optimal weight vector θ^* that fits our data and *we posit that network architectures should be characterised by how easy it is to estimate their optimal network weights* θ^*. We build upon statistical learning theory and use Fisher Information [28] framework to formally describe expected behaviour of the training process of a given deep network f.

The Fisher Information Matrix (FIM) encompasses information on the difficulty of the parameter estimation problem and it plays a crucial role in several fundamental results, which we apply below in the context of deep networks. The FIM of a network f and its set of weights $\theta \in \mathbb{R}^p$ is given as

$$F(\theta) \coloneqq \mathbb{E}\left[\nabla_\theta \sigma_\theta(c \,|\, x)\, \nabla_\theta \sigma_\theta(c \,|\, x)^T\right] \in \mathbb{R}^{p \times p}, \quad (2)$$

where we take the expected value $\mathbb{E}$ with respect to the joint distribution of (x, c). For more detailed account on Fisher Information theory and its applications in the context of machine learning, we kindly refer reader to [16, 20, 29, 35, 36].

Cramér-Rao bound. The first part of estimation theory we build upon is the inverse of the FIM, known as the Cramér–Rao bound (see [11]), which is the asymptotic variance of the estimated weights (i.e. the uncertainty coming from the data variability). Thus, the larger the matrix norm of the FIM, the more certain we are about the weights θ. More specifically, any data-dependent estimator $\hat{\theta}_n = \hat{\theta}_n(x_1, \ldots, x_n)$ is a random vector with randomness coming from the (independent) choice of input images $x_1, \ldots, x_n$ and as such has some variance matrix $\mathrm{Var}\left(\hat{\theta}_n\right)$. The famous result (see [8, 37]) named in honour of H. Cramér and C. R. Rao states that if $\hat{\theta}_n$ is unbiased then the variance is bounded from below as

$$\mathrm{Var}\left(\hat{\theta}_n\right) \geq \frac{1}{n} F^{-1}(\theta) \quad (3)$$

and for maximum likelihood estimation this bound is attained as the number of input images grows to infinity $n \to \infty$.

We formally show (see Supplementary material) that for each weight $\theta(j)$ the mean square error of our estimation is controlled by the diagonal element of the FIM inverse as

$$\mathbb{E}\left(\hat{\theta}_n(j) - \theta(j)\right)^2 \geq \frac{1}{n}\left(F^{-1}(\theta)\right)_{jj}. \tag{4}$$

Therefore, knowing the FIM allows us to evaluate how certain we are about the weight estimates. We can go even further by inspecting the eigenvalues of $F(\theta)$. Denoting the largest and smallest eigenvalue of the FIM as $\lambda_{\min}$ and $\lambda_{\max}$ respectively, then we have that there exist linear combination coefficients $e_{\min}$ and $e_{\max}$ of unit size so that

$$\begin{aligned}\mathbb{E}\left(e_{\min}^T\hat{\theta}_n - e_{\min}^T\theta\right)^2 \geq & \frac{1}{n\lambda_{\min}},\\ \mathbb{E}\left(e_{\max}^T\hat{\theta}_n - e_{\max}^T\theta\right)^2 \geq & \frac{1}{n\lambda_{\max}},\end{aligned} \tag{5}$$

indicating that if the difference between the largest and smallest eigenvalue is *large* then there exist combinations of weights with *very large difference* in the estimation certainty. Altogether, the more the eigenvalues of the FIM are similar, the more similar is also the variance in the weight estimation across all model weights.

Prediction sensibility. The change of the model prediction measured by the KL-divergence when subject to a small perturbations of the weights is given as

$$D_{KL}(\sigma_{\theta+\theta_\delta}(\cdot\,|\,x), \sigma_\theta(\cdot\,|\,x)) \approx \frac{1}{2}\theta_\delta^T\, F(\theta)\, \theta_\delta \tag{6}$$

for a small weight perturbation vector $\theta_\delta \in \mathbb{R}^p$. Thus, $F(\theta)$ measures volatility of the predictions subject to the weight change – if the difference between the largest and the smallest eigenvalues of the FIM is large, then there exist perturbation directions $\theta_{min}, \theta_{max}$ corresponding to $\lambda_{min}, \lambda_{max}$ such that

$$\begin{aligned}D_{KL}(\sigma_{\theta+\theta_{min}}(\cdot\,|\,x), \sigma_\theta(\cdot\,|\,x)) &\approx \frac{1}{2}\theta_{min}^T\, F(\theta)\, \theta_{min} = \\ &= \frac{1}{2}\lambda_{min}\|\theta_{min}\|^2 \ll \frac{1}{2}\lambda_{max}\|\theta_{max}\|^2 = \\ &= \frac{1}{2}\theta_{max}^T\, F(\theta)\, \theta_{max} \approx D_{KL}(\sigma_{\theta+\theta_{max}}(\cdot\,|\,x), \sigma_\theta(\cdot\,|\,x))\end{aligned} \tag{7}$$

and therefore in some directions a small change of the weights has much larger impact on the prediction than in others, making the model less balanced. For further discussion, see the Supplementary material.

2.2. Empirical Fisher Information Matrix implementation

When independent and identically distributed sample images x_n are available, empirical FIM $\hat{F}(\theta)$ is defined as

$$\hat{F}(\theta) \coloneqq \frac{1}{n}\sum_{n=1}^{N} \mathbb{E}_{\sigma_\theta}\left[\nabla_\theta \sigma_\theta(c\,|\,x_n)\, \nabla_\theta \sigma_\theta(c|\,x_n)^T\right] \tag{8}$$

where $\mathbb{E}_{\sigma_\theta}$ now denotes the expectation with respect to the model prediction σ_θ.

We would like to first emphasize several crucial aspects and *contributions of this paper* that lead to the first success of the Fisher Information theory (having e.g. [1] in mind) in the context of Neural Architecture Search, despite Fisher Information being one of the first obvious choices for deep network analysis:

1. Following [19] we write the empirical FIM (8) as

$$\begin{aligned}\hat{F}(\theta) = \frac{1}{n}\sum_{n=1}^{N}\big[\nabla_\theta\Psi(x_n,\theta)^T\,(\mathrm{diag}(\sigma_\theta(\cdot,x_n)) - \\ \sigma_\theta(\cdot,x_n)\sigma_\theta(\cdot,x_n)^T)\,\nabla_\theta\Psi(x_n,\theta)\big].\end{aligned} \tag{9}$$

and we further decompose the inner matrix $\mathrm{diag}(\sigma_\theta(\cdot,x_n)) - \sigma_\theta(\cdot,x_n)\sigma_\theta(\cdot,x_n)^T$ using analytical formulas from [40] to avoid numerical instability of the computation as we arrive at a feasible representation

$$\hat{F}(\theta) = \frac{1}{n}\sum_{n=1}^{N} A_n^T A_n \tag{10}$$

for some matrices $A_n \in \mathbb{R}^{C\times p}$. We observed that networks typically yield very imbalanced outputs at initialization (i.e. every network prioritizes few classes over the rest) making it extremely important not to exclude the factor $\mathrm{diag}(\sigma_\theta(\cdot,x_n)) - \sigma_\theta(\cdot,x_n)\sigma_\theta(\cdot,x_n)^T$, which is however difficult to compute without underflow/overflow.

2. The dimension of FIM is equal to p (the number of all trainable parameters) and therefore direct computation of the eigenvalues is numerically intractable and unstable. However, our results show that if a small number of representative parameters is drawn, then computation becomes stable while the discrimination power does not suffer. We used a simple rule where a single parameter from each trainable layer (not including batch normalization) is chosen. Stability with respect to choice of such parameters can be found in Supplementary material.

3. FIM of large networks typically suffers from having pathological spectrum, i.e. there usually exist zero eigenvalue and its multiplicity is large (see [15]) and thus the eigenvalues estimation due to a large condition number is imprecise. However, as we deal with a symmetric positive-semidefinite matrix, the eigenvalues actually coincide with the singular values (see [24]), for which the estimation algorithm performs better.

Let us also emphasize that there is a common misconception in part of the community as it is often mistakenly assumed that one can simply use the true labels of x_n in (8) in place of c. However, such definition is then meaningless as it does not approximate the FIM in the classical Monte Carlo sense (see Supplementary material and [19] for a comparison). That means that the empirical FIM *does not* depend on the true labels and therefore our method *does not* require real data - we use random input instead.

2.3. Variance of Knowledge for Deep Network Weights

In order to characterize properties of the parameter estimation process for a given network f through the lens of Fisher Information theory, we inspect the eigenvalues of the empirical FIM $\hat{F}(\theta_{\text{init}})$ and define the entropy of *Variance of Knowledge for Deep Network Weights (VKDNW)* as

$$\text{VKDNW}\big(f\big) \coloneqq -\sum_{k=1}^{9} \tilde{\lambda}_k \log \tilde{\lambda}_k$$
$$\tilde{\lambda}_k = \frac{\lambda_k}{\sum_{j=1}^{9} \lambda_j}, \quad k = 1, \ldots, 9, \tag{11}$$

where λ_k denotes the k-th decile of the FIM eigenvalues as the representation of the FIM spectrum[1], and θ_{init} denotes network weights at initialization.

Our score therefore measures the diversity of the FIM eigenvalues, and from the entropy theory we know that VKDNW attains its maximum exactly when all the eigenvalues λ_k are equal and VKDNW gets lower as the eigenvalues become *more different*. Based on the discussion in Sec. 2.1 we see that VKDNW is high when the uncertainty in all model weight combinations are similar (see Cramér-Rao bound) and there are no directions in the weight space that would influence the network prediction substantially differently than others. Due to the fact that we have normalized both the number of eigenvalues under consideration (by taking a fixed number of representatives irrespective of the network size) and the magnitude of the eigenvalues, VKDNW is independent of network size (number of network weights p).

Let us note, that we are familiar with the fact that even though our motivation was (among others) based on Cramér-Rao bound that assumes evaluation of FIM at the *correct* weight vector θ_X that fits the data, which is typically far away from the weights given at initialization. However, our empirical results below support the hypothesis that the evaluation despite being in the wrong point brings valuable information.

Ranking networks for NAS. The proposed VKDNW score is independent of network size, which is extremely beneficial to compare individual structures of network architectures. Thus, it does not aim to capture capacity, rather it targets feasibility of the computation graph given number of operations. However, when comparing different network structures and different network sizes together as it is done in NAS, one indeed needs to take network size into account as well, because naturally larger networks have bigger capacity and therefore tend to have higher accuracy. To capture also the capacity we proxy the network size by the number of layers with weights that we denote as $\aleph\big(f\big)$ for a network f and we introduce the ranking

$$\text{VKDNW}_{\text{single}}\big(f\big) \coloneqq \aleph\big(f\big) + \text{VKDNW}\big(f\big) \tag{12}$$

Here we leverage the fact that VKDNW as an entropy of some quantity is always between 0 and 1 and by summing it with an integer-valued quantity we in fact obtain that we have first grouped networks by our size proxy $\aleph$ and then within each group of similar networks sizes we order them by VKDNW.

[1] As explained in Sec. 2.2, the smallest eigenvalue λ_0 is usually equal to zero and thus we exclude it for stability reasons (similarly with the maximal eigenvalue λ_{10})

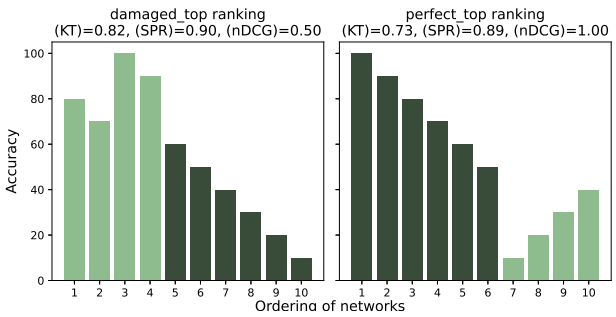

Figure 2. Toy example of two rankings on 10 networks. We plot accuracies ordered by the rankings and evaluation metrics Kendall's τ (KT) and Spearman's ρ (SPR) correlations and Normalized Discounted Cumulative Gain (nDCG_5).

3. Evaluation Metrics

In this section, we dive into the problem of evaluating (image) classification accuracy proxies for training-free NAS (TF-NAS). In NAS algorithms, the proxy actually does not have to predict the classification accuracy in absolute terms – since we are after finding the "best" network for given task – it's sufficient that the proxy properly ranks the networks, ordering them from worst to the best.

Suppose that a collection of K networks with validation accuracies $\text{acc}_1, \ldots, \text{acc}_K$ have been ranked by a TF-NAS proxy as $r_1, \ldots, r_K \in \mathbb{R}$. The standard way of evaluating [14, 21, 25] how well the ranks correspond to the accuracies is to compute Kendall's τ (KT, [17]) and Spearman's ρ (SPR, [39]) rank correlation coefficients. Kendall's τ is given as

$$\tau \coloneqq \frac{n_c - n_d}{\sqrt{n_c + n_d + n_1}\sqrt{n_c + n_d + n_2}}, \tag{13}$$

where n_c denotes the number of concordant pairs (acc_k, r_k), n_d the number of discordant pairs and n_1 (resp. n_2) denote the number of ties in acc_k only (resp. in r_k only). In our context, (KT) relates after some normalization to a probability that two randomly chosen network rankings r_k, r_l are ordered correctly according to their accuracies $\text{acc}_k, \text{acc}_l$. In contrast, (SPR) is defined to account more for outliers, i.e. it tends to put a large penalization if

there are some networks for which the difference between the rank of the accuracy acc_k and the rank r_k is large as it is just the classical correlation coefficient however applied on the orders of the assessed quantities.

Normalized Discounted Cumulative Gain. Both (KT) and (SPR) are good evaluation metrics when our interest lies in comparison of ranking proxies when *all networks involved are of the same importance*. However, in NAS we are primarily interested if a ranking helps us to pick best networks from a given collection and less interested how well the ranking compares two networks with low accuracy. Finding inspiration in information retrieval where one measures the quality of a system that retrieves resources relevant to an input query, we propose to use Normalized Discounted Cumulative Gain (nDCG) [2] as a more relevant metric to measure quality of TF-NAS proxies. The nDCG metric is defined with a key requirement that highly relevant documents are more valuable when they appear earlier in the search engine results (i.e., in higher-ranking positions). This requirement can be reformulated for the task of neural architecture search as *networks with high accuracies are more valuable when they appear on higher-ranking positions*.

We thus define the metric as follows: first we order the networks by their ranking $r_1, \dots, r_K$ so that $r_{k_1} = K, r_{k_2} = K-1, \dots$ and we compute

$$\text{nDCG}_P \coloneqq \frac{1}{Z}\left(\sum_{j=1}^{P} \frac{2^{\text{acc}_{k_j}} - 1}{\log_2(1+j)}\right), \tag{14}$$

where $P \in \mathbb{N}$ is a parameter determining how many top-ranked networks do we consider (e.g. it corresponds to the population size in an evolution algorithm) and Z is a normalization factor that represents the ideal discounted cumulative gain so that nDCG is equal to one for a perfect fit.

The higher nDCG the better the ranking is as it is a weighted average of transformed top-ranked accuracies. In [47] it is shown that the choice of the discount factor of given by inverse logarithm is a good choice as it nDCG then well separates different ranking systems. [2]

Toy Example. Let us briefly demonstrate the weak ability of (KT) and (SPR) to distinguish rankings that poorly discriminate networks with high accuracy. Suppose a collection of 10 networks with their validation accuracies $100, 90, \cdots, 10$ is given and our aim is to rank them from worst to the best. We evaluate two different toy rankings: a) ranking `damaged_top` which perfectly fits the accuracies, only the two top networks are swapped with the third and fourth, b) ranking `perfect_top` which fits perfectly for the top-performing networks, however, now the order of the four worst networks is reversed (see Figure 2). The `damaged_top` ranking is an example that should be evaluated worse for the architecture search task compared to `perfect_top` as in the first case the best networks are not placed first, while in the second they are. On the other hand, even though `perfect_top` is not a perfect fit, it still discriminates the top networks ideally and o*nly struggles for networks with low accuracy*, which are not interesting for NAS which is after the best networks. From the correlation perspectives of (KT), (SPR) `damaged_top` is better than `perfect_top`, therefore, if we rely only on these two evaluation metrics we prefer ranking that is worse for the architecture search task as the top networks are ranked worse. On the other hand, for `perfect_top` we obtain ideal nDCG_5 while it drops to 0.5 for `damaged_top`. We conclude that using nDCG_5 we choose ranking that has higher discriminatory power for good networks.

We also note that from statistical perspective the (KT) and (SPR) of `damaged_top` is significantly higher than of a random ranking[3], and therefore we'd conclude that `damaged_top` is not independent of ground truth accuracy. On the other hand, when we perform the same statistical test on `damaged_top` using nDCG_5, we conclude that the `damaged_top` is not significantly better than ranking networks randomly[4].

4. Related Work

Zero-shot NAS aims to rank given networks in a training-free manner based on their (a priori unknown) final performance, which allows to prune the huge search space with limited costs when seeking for optimal architecture [10, 27, 46] and other configuration [3, 26, 38].

Throughout previous works different approaches for ranking computation can be found. In many of the works the gradient with respect to the network weights is investigated (GradNorm, GraSP, SNIP, Synflow, [1, 23, 41, 42]), i.e. leveraging the first order approximation of the network. In Jacov [1] correlations of the Jacobian matrices among various input samples are compared; in NASWOT [31] the linear maps induced by data points are examined; Zen [26] uses the approximation of gradient with respect to featuremaps; GradSign [49] compares the optimization landscape at the level individual training samples; or in ZiCo [25] the gradients from multiple forward and backward passes preferring large magnitude and low variance.

Other methods aggregate multiple sources aiming at obtaining a better informed ranks: TE-NAS [6] uses both the number of linear regions [12, 44] and the condition number of Neural Tangent Kernel [13, 22]. However, it is well

[2] In case of ties we take random ordering within the groups (parameter `ignore_ties=True` in the scikit-learn implementation).

[3] That is, the p-value for the hypothesis that `damaged_top` is assigned independently of the accuracies is below 0.005.

[4] Running 1000 samples on a random ranking we obtain that `damaged_top` has nDCG_5 around 75th percentile of such random evaluations and it is not significantly better than a random ranking.

	Type	CIFAR-10			CIFAR-100			ImageNet16-120		
		KT	SPR	nDCG	KT	SPR	nDCG	KT	SPR	nDCG
Simple rankings										
FLOPs	S	0.623	0.799	0.745	0.586	0.763	0.576	0.545	0.718	0.403
GradNorm [1]	S	0.328	0.438	0.509	0.341	0.451	0.278	0.310	0.418	0.265
GraSP [1, 42]	S	0.352	0.505	0.518	0.349	0.498	0.284	0.359	0.502	0.281
SNIP [1, 23]	S	0.431	0.591	0.513	0.440	0.597	0.286	0.389	0.521	0.286
SynFlow [1, 41]	S	0.561	0.758	0.709	0.553	0.750	0.594	0.531	0.719	0.511
Jacov [1]	S	0.616	0.800	0.540	0.639	0.820	0.402	0.602	0.779	0.356
NASWOT [31]	S	0.571	0.762	0.607	0.607	0.799	0.475	0.605	0.794	0.490
ZenNAS [26]	S	0.102	0.103	0.120	0.079	0.072	0.115	0.091	0.109	0.073
GradSign† [49]	S	·	0.765	·	·	0.793	·	·	0.783	·
ZiCo [25]	S	0.607	0.802	0.751	0.614	0.809	0.607	0.587	0.779	0.523
TE-NAS [6]	A	0.536	0.722	0.602	0.537	0.723	0.327	0.523	0.709	0.330
AZ-NAS [21]	A	0.712	0.892	0.749	0.696	0.880	0.549	0.673	0.859	0.534
No. of trainable layers ($\aleph$)	S	0.626	0.767	0.671	0.646	0.787	0.525	0.623	0.764	0.497
$\text{VKDNW}_{\text{single}}$ (ours)	S	0.618	0.815	0.751	0.634	0.829	0.617	0.622	0.814	0.608
$\text{VKDNW}_{\text{agg}}$ (ours)	A	**0.750**	**0.919**	**0.785**	**0.753**	**0.919**	**0.636**	**0.743**	**0.906**	**0.664**
Model-driven rankings										
GRAF [14]	A	0.820	0.953	0.935	0.809	0.948	0.858	0.796	0.941	0.828
VKDNW_m (ours)	A	0.647	0.831	0.750	0.636	0.821	0.602	0.611	0.798	0.575
$(\text{VKDNW+ZCS})_m$ (ours)	A	0.840	0.963	0.922	0.834	0.960	0.884	0.830	0.958	0.843
$(\text{VKDNW+ZCS+GRAF})_m$ (ours)	A	**0.859**	**0.971**	**0.946**	**0.847**	**0.966**	**0.895**	**0.842**	**0.963**	**0.867**

Table 1. Training-free NAS methods in the NAS-Bench-201 [10] search space, evaluated on three public datasets. Kendall's τ (KT), Spearman's ρ (SPR) and Normalized Discounted Cumulative Gain (nDCG) are reported, results are averages of 5 independent runs (all stds are < 0.02). The Type column differentiates single (S) and aggregated (A) rankings. Results are reproduced with code published by their authors, except those marked†, where results from the original paper are taken.

known that the kernel computation is highly computationally demanding [34]. AZ-NAS [21] assesses expressivity, trainability and progressivity via examination of feature distribution across all orientations and the Jacobian. However, despite AZ-NAS outperforming previous works in some metrics, it is worse than [25] in key NAS-related aspects such as the cumulative gain (see Sec. 5.1).

Despite a tremendous effort of the community, it was shown that most of the zero-shot NAS methods perform worse than a simple proxy given just by FLOPs or #params [33, 43]. Thus, there is still a gap for improvements also driven by the need of the ranking explainability that would have a satisfactory theoretical support.

Furthermore, [14] uses a model-driven ranking that however needs a set of networks for which the validation accuracies are known to train the model and the ability of this ranking to generalize as it is fitted on a specific network collection only is disputable.

Finally, let us discuss the current practice in the NAS methods comparison. Methods are compared either by means of correlations of their scores to accuracy or by reporting the accuracy of the top-ranked network [21, 25]. However, while the first is not tailor-made for the architecture search task and therefore does not assess the desired ranking properties (see Sec. 5.1), comparing performance just by the accuracy of a single network is too vulnerable e.g. to the random seed choice.

5. Experiments

5.1. Ranking aggregation

In addition to using the single ranking of Eq. (12), we also experiment with multiple rankings (as in [14, 21, 25]) to order network architectures by their accuracy. We use two different options: non-linear and model-driven aggregation.

Non-linear aggregation. The aggregation [21] uses multiplication to combine multiple ranks into a single one, which means a network is highly-ranked if and only if it is highly-ranked in *all* subsidiary rankings, keeping their influence balanced. That is, denoting rankings $\text{rank}_1, \ldots, \text{rank}_m$ for some $m \in \mathbb{N}$ we define the aggregated ranking

$$\text{rank}_{\text{agg}}(f) := \log \Pi_{j=1}^{m} \text{rank}_j(f) \tag{15}$$

for each network f. This aggregation is therefore possible only in the context of a given network collection, such as in evolutionary search.

In our case, $\text{VKDNW}_{\text{agg}}$ aggregates these five proxies:

- $\text{VKDNW}_{\text{single}}$ (V) ranking (Eq. (12)),
- Jacov (J) [1] measures the activations correlation when

Method	FLOPs	Top-1 acc.	Type	Search cost (GPU days)
NASNet-B [50]	488M	72.8	MS	1800
CARS-D [45]	496M	73.3	MS	0.4
BN-NAS [5]	470M	75.7	MS	0.8
OFA [4]	406M	77.7	OS	50
RLNAS [48]	473M	75.6	OS	-
DONNA [32]	501M	78.0	OS	405
# Params	451M	63.5	ZS	0.02
ZiCo [25]	448M	78.1	ZS	0.4
AZ-NAS [21]	462M	78.6	ZS	0.4
$\text{VKDNW}_{\text{agg}}$ (ours)	480M	**78.8**	ZS	0.4

Table 2. Results on ImageNet-1K [9] in the MobileNetV2 search space, the size of the model is constrained to ≈450M FLOPS.

exposed to various inputs,

- Expressivity (E) [21] assesses isotropy and uniformity of the features distribution across all orientations,
- Trainability (T) [21] captures ability of the network to keep stable gradient propagation between the layers by inspecting the spectrum of the Jacobian matrix,
- FLOPs (F) is the number of FLOPs for one forward pass.

Model-driven aggregation. When accuracies of a sufficient number of architectures in the search space are known, model-driven aggregation can be used to train a regression model to combine individual rankings. The trained model is then used to predict accuracy for unseen networks in the same search space. We evaluate three different models:

- VKDNW_m where the eigenvalues λ_k of the FIM matrix are used directly as features in companion with $\aleph$ (number of trainable layers) and FLOPs to allow a more complex proxy of the diversity of the eigenvalues and the network complexity than the simple entropy Eq. (11),
- $(\text{VKDNW+ZCS})_m$ where we additionally include all other zero-cost scores available (see Table 1),
- $(\text{VKDNW+ZCS+GRAF})_m$ where we add network graph features from [14].

5.2. Results

We have conducted experiments in the NAS-Bench-201 [10] and MobileNetV2 [26, 38] search spaces. To obtain easily comparable results, we used 64 randomly generated input images to compute our score as in [21]. For the methods that rely on the knowledge of true labels, input data from the respective datasets were used.

NAS-Bench-201. The dataset consists of 15,625 networks for which validation accuracies for CIFAR-10, CIFAR-100 [18] and ImageNet16-120 [7] after training for 200 epochs are provided. The networks are characterized by unique cell structures comprising of several types of operation choices. As one of the possible choices is zero operation, it's possible that some computation edges don't receive any input or cannot propagate their results to the output, leading to same computation graphs for different architectures, thus duplicating networks. Following the practice from NAS-Bench-101 [46], we report results on 9,445 unique structures (also in [14, 30]) and we refer the reader to Supplementary material for results on all networks. We measure proxies performance via Kendall's τ (KT) and Spearman's ρ (SPR) correlations with validation accuracies together with Normalized Discounted Cumulative Gain (nDCG_{1000}, we write nDCG for short) from Sec. 3, averaged over 5 independent runs. For the model-driven aggregation, we trained a random forest model on 1024 networks for 100 iterations, and used the rest for testing.

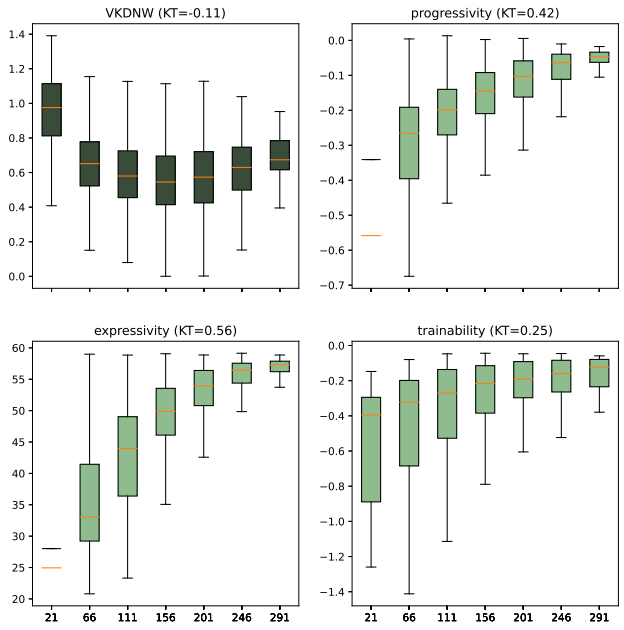

Figure 3. Components of AZ-NAS [21] and our VKDNW are compared w.r.t. correlation with $\aleph$ (number of trainable layers), in the NAS-Bench-201 search space [10] on ImageNet16-120 [7] dataset. Our VKDNW proxy has the lowest correlation, ie. is the most invariant to the size of the model.

In Table 1, we compare our method with existing NAS methods and show that $\text{VKDNW}_{\text{agg}}$ outperforms all methods with a significant margin in all three metrics. The benefit of our method is two-fold: a) it identifies the best networks due to its high performance in nDCG, which is the desirable property in NAS b) the ranking is consistent across the whole network search space due to high KT and SPR correlations. Note that we can only make these observations by combining standard correlation metrics and our newly proposed nDCG. We also show that $\text{VKDNW}_{\text{single}}$ outperforms all other single-rank proxies in all metrics on ImageNet16-120, in SPR on all three datasets, while being among the highest on CIFAR-10 and CIFAR-100.

MobileNetV2. We search for the best network configuration in the MobileNetV2 space [38], while constraining

V	J	E	T	F	(KT)	(SPR)	nDCG
✓					**0.622**	**0.814**	**0.608**
	✓				0.603	0.781	0.339
		✓			0.588	0.779	0.274
			✓		0.353	0.517	0.233
				✓	0.545	0.718	0.403
✓	✓	✓	✓		0.717	0.891	0.623
✓	✓	✓		✓	0.706	0.871	0.553
✓	✓		✓	✓	**0.736**	**0.905**	**0.695**
✓		✓	✓	✓	0.722	0.896	0.658
	✓	✓	✓	✓	0.735	0.901	0.646
✓	✓	✓	✓	✓	**0.743**	**0.906**	**0.664**

Table 3. Components of $\text{VKDNW}_{\text{agg}}$ rank with non-linear aggregation. Consistency is shown with respect to Kendall's τ (KT), Spearman's ρ (SPR) and Normalized Discounted Cumulative Gain (nDCG) with $P = 1000$ on ImageNet16-120 image dataset [7]. Here V, J, E, T and F stand for VKDNW, Jacov, expressivity, trainability and FLOPs respectively (see Sec. 5.1). Table of all combinations can be found in the Supplementary material.

the model size to approximately 450M FLOPs. We ran 100,000 iterations of the evolutionary search algorithm [21] for approximately 10 hours, meaning 100,000 different architectures were evaluated. We then took the best network from the search and trained it for 480 epochs on ImageNet-1K [9], using the same hyper-parameter setting as in [21, 25]. The final training of the model took 7 days on 8xNVidia A100 GPUs.

As seen in Table 2, our method $\text{VKDNW}_{\text{agg}}$ outperforms all prior approaches – even train-based approaches (denoted MS and OS) that incur much higher computational costs for the search.

5.3. Ablations

We ablate our method in the NAS-Bench-201 search space [10] on ImageNet16-120 [7] validation accuracies.

Orthogonality of VKDNW. Our score VKDNW has a desirable property that it is based on information orthogonal to the size of the network: in Figure 3 we can see that unlike previous work, VKDNW is not correlated with the network size measured by $\aleph$ (number of trainable layers). We believe this property is key when stepping into much larger search spaces, however components of the previous state-of-the-art method AZ-NAS lack such a property (also Fig. 3 and Tab. 4). We conjecture that improvement in the key metrics by VKDNW is caused by this orthogonality feature. In Table 1 we also provide comparison to previous model-driven method and show that adding our feature significantly improves the ranking in all considered metrics.

Components of aggregated rank. Our single-rank variant $\text{VKDNW}_{\text{single}}$ is the strongest component of the aggregated $\text{VKDNW}_{\text{agg}}$ as can be seen in Table 3 where it outperforms rest of the rankings with especially large margin in nDCG, thus, it strongly discriminates high-accuracy networks compared to others.

	(KT)		
	CIFAR-10	CIFAR-100	ImageNet16-120
VKDNW	0.041	-0.090	-0.107
trainability	0.220	0.220	0.253
expressivity	0.539	0.539	0.539
progressivity	0.398	0.398	0.385

Table 4. Kendall's τ (KT) of $\aleph$ (number of trainable layers) with components of AZ-NAS compared to our new method VKDNW on three datasets on NAS-Bench-201 search space [10] and multiple image datasets.

Batch Size	Random Input			Real Input		
	KT	SPR	nDCG	KT	SPR	nDCG
8	0.615	0.808	0.617	0.614	0.805	0.599
16	0.614	0.808	0.604	0.608	0.800	0.595
32	0.619	0.812	0.611	0.612	0.804	0.594
64	0.621	0.814	0.617	0.614	0.806	0.595
128	0.621	0.814	0.616	0.615	0.806	0.591

Table 5. Our method VKDNW evaluated for different batch sizes using either randomly generated input data or real images with respect to Kendall's τ (KT), Spearman's ρ (SPR) and Normalized Discounted Cumulative Gain with $P = 1000$ (nDCG).

Random or real input. As our method is computed using generated random data (white noise) it does not rely on any real dataset and is therefore applicable also in situations where no reliable data are available. Table 5 shows that by relying just on random data we do not lose any performance with respect to all key metrics. Moreover, the performance remains relatively stable across different batch sizes. We chose batch size 64 as larger batch size does not bring better results.

Due to lack of space, we kindly refer the reader to Supplementary material for further ablations.

6. Conclusion

We proposed a new training-free NAS proxy called *Variance of Knowledge of Deep Network Weights (VKDNW)*. The method has strong theoretical support and achieved state-of-the-art results on three public datasets and in two search spaces.

Based on throughout evaluation and comparison to other existing approaches, it has been shown that it provides an information orthogonal to the network size leading to a zero-cost ranking were contribution of the network size and architecture feasibility are separated. We have also shown that previously used correlation metrics for proxy evaluations do not sufficiently assess the key ability to discriminate top networks and to address this problem we have proposed a new evaluation metric and re-evaluated previous methods with the new metric.

Acknowledgement

The research was supported by Czech Science Foundation Grant No. 24-10738M. The access to the computational infrastructure of the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics" and of the Ministry of Education, Youth and Sports of the Czech Republic project e-INFRA CZ (ID:90254) is also gratefully acknowledged.

References

[1] Mohamed S Abdelfattah, Abhinav Mehrotra, Łukasz Dudziak, and Nicholas D Lane. Zero-cost proxies for lightweight NAS. *arXiv preprint arXiv:2101.08134*, 2021.

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96, 2005.

[3] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

[4] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019.

[5] Boyu Chen, Peixia Li, Baopu Li, Chen Lin, Chuming Li, Ming Sun, Junjie Yan, and Wanli Ouyang. BN-NAS: Neural architecture search with batch normalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 307–316, 2021.

[6] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. *arXiv preprint arXiv:2102.11535*, 2021.

[7] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

[8] Harald Cramér. *Mathematical methods of statistics*. Princeton university press, 1999.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Xuanyi Dong and Yi Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv preprint arXiv:2001.00326*, 2020.

[11] Roy Frieden and Robert A Gatenby. *Exploratory data analysis using Fisher information*. Springer Science & Business Media, 2010.

[12] Boris Hanin and David Rolnick. Complexity of linear regions in deep networks. In *International Conference on Machine Learning*, pages 2596–2604. PMLR, 2019.

[13] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[14] Gabriela Kadlecová, Jovita Lukasik, Martin Pilát, Petra Vidnerová, Mahmoud Safari, Roman Neruda, and Frank Hutter. Surprisingly strong performance prediction with neural graph features. *arXiv preprint arXiv:2404.16551*, 2024.

[15] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. Pathological spectra of the fisher information metric and its variants in deep neural networks. *arXiv preprint arXiv:1910.05992*, 2019.

[16] Ryo Karakida, Shotaro Akaho, and Shun-ichi Amari. Universal statistics of fisher information in deep neural networks: Mean field approach. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1032–1041. PMLR, 2019.

[17] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1-2):81–93, 1938.

[18] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[19] Frederik Kunstner, Philipp Hennig, and Lukas Balles. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems*, 32, 2019.

[20] Byung-Kwan Lee, Junho Kim, and Yong Man Ro. Masking adversarial damage: Finding adversarial saliency for robust and sparse network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15126–15136, 2022.

[21] Junghyup Lee and Bumsub Ham. AZ-NAS: Assembling zero-cost proxies for network architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5893–5903, 2024.

[22] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

[23] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

[24] Steven J Leon, Lisette De Pillis, and Lisette G De Pillis. *Linear algebra with applications*. Pearson Prentice Hall Upper Saddle River, NJ, 2006.

[25] Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. ZiCo: Zero-shot NAS via inverse coefficient of variation on gradients. In *The Eleventh International Conference on Learning Representations*, 2023.

[26] Ming Lin, Pichao Wang, Zhenhong Sun, Hesen Chen, Xiuyu Sun, Qi Qian, Hao Li, and Rong Jin. Zen-nas: A zero-shot nas for high-performance image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 347–356, 2021.

[27] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[28] Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul PPP Grasman, and Eric-Jan Wagenmakers. A tutorial on fisher information. *Journal of Mathematical Psychology*, 80:40–55, 2017.

[29] James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21 (146):1–76, 2020.

[30] Abhinav Mehrotra, Alberto Gil CP Ramos, Sourav Bhattacharya, Łukasz Dudziak, Ravichander Vipperla, Thomas Chau, Mohamed S Abdelfattah, Samin Ishtiaq, and Nicholas Donald Lane. Nas-bench-asr: Reproducible neural architecture search for speech recognition. In *International Conference on Learning Representations*, 2021.

[31] Joe Mellor, Jack Turner, Amos Storkey, and Elliot J Crowley. Neural architecture search without training. In *International conference on machine learning*, pages 7588–7598. PMLR, 2021.

[32] Bert Moons, Parham Noorzad, Andrii Skliar, Giovanni Mariani, Dushyant Mehta, Chris Lott, and Tijmen Blankevoort. Distilling optimal neural networks: Rapid search in diverse spaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12229–12238, 2021.

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

[34] Roman Novak, Jascha Sohl-Dickstein, and Samuel S Schoenholz. Fast finite width neural tangent kernel. In *International Conference on Machine Learning*, pages 17018–17044. PMLR, 2022.

[35] Hyeyoung Park and Kwanyong Lee. Adaptive natural gradient method for learning of stochastic neural networks in mini-batch mode. *Applied Sciences*, 9(21):4568, 2019.

[36] Jeffrey Pennington and Pratik Worah. The spectrum of the fisher information matrix of a single-hidden-layer neural network. *Advances in neural information processing systems*, 31, 2018.

[37] C Radhakrishna Rao. Information and the accuracy attainable in the estimation of statistical parameters. In *Breakthroughs in Statistics: Foundations and basic theory*, pages 235–247. Springer, 1992.

[38] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[39] Charles Spearman. The proof and measurement of association between two things. 1961.

[40] Kunio Tanabe and Masahiko Sagae. An exact cholesky decomposition and the generalized inverse of the variance–covariance matrix of the multinomial distribution, with applications. *Journal of the Royal Statistical Society: Series B (Methodological)*, 54(1):211–219, 1992.

[41] Hidenori Tanaka, Daniel Kunin, Daniel L Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *Advances in neural information processing systems*, 33:6377–6389, 2020.

[42] Chaoqi Wang, Guodong Zhang, and Roger Grosse. Picking winning tickets before training by preserving gradient flow. *arXiv preprint arXiv:2002.07376*, 2020.

[43] Colin White, Mikhail Khodak, Renbo Tu, Shital Shah, Sébastien Bubeck, and Debadeepta Dey. A deeper look at zero-cost proxies for lightweight NAS. *ICLR Blog Track*, 2022.

[44] Huan Xiong, Lei Huang, Mengyang Yu, Li Liu, Fan Zhu, and Ling Shao. On the number of linear regions of convolutional neural networks. In *International Conference on Machine Learning*, pages 10514–10523. PMLR, 2020.

[45] Zhaohui Yang, Yunhe Wang, Xinghao Chen, Boxin Shi, Chao Xu, Chunjing Xu, Qi Tian, and Chang Xu. CARS: Continuous evolution for efficient neural architecture search. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1829–1838, 2020.

[46] Chris Ying, Aaron Klein, Eric Christiansen, Esteban Real, Kevin Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *International conference on machine learning*, pages 7105–7114. PMLR, 2019.

[47] Wang Yining, Wang Liwei, Li Yuanzhi, He Di, Chen Wei, and Liu Tie-Yan. A theoretical analysis of ndcg ranking measures. In *Proceedings of the 26th annual conference on learning theory*, 2013.

[48] Xuanyang Zhang, Pengfei Hou, Xiangyu Zhang, and Jian Sun. Neural architecture search with random labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10907–10916, 2021.

[49] Zhihao Zhang and Zhihao Jia. Gradsign: Model performance inference with theoretical insights. *arXiv preprint arXiv:2110.08616*, 2021.

[50] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.