L-SWAG: Layer-Sample Wise Activation with Gradients information for Zero-Shot NAS on Vision Transformers

Supplementary Material

A. Proof of Theorem 1

Theorem 1. *Given a linear regressor $f(\boldsymbol{a}, \boldsymbol{x})$ with trainable parameters $\boldsymbol{a} = (a_j)_{j=1}^M$, let $g(\boldsymbol{x}_i) = (g_j(\boldsymbol{x}_i))_{j=1}^d$ be the gradient of $\boldsymbol{a}$ w.r.t. to $\boldsymbol{x}_i$, and $\hat{\boldsymbol{a}} = \boldsymbol{a} - \eta \sum_i g_j(\boldsymbol{x}_i)$ the updated parameters with learning rate η. Denote $\mu_j = \frac{1}{M}\sum_i g_j(\boldsymbol{x}_i)$, $\sigma_j = \sqrt{\sum_i (g_j(\boldsymbol{x}_i) - \mu_j)^2}$. Then, for any η, the total training loss $\mathcal{L}_f(\boldsymbol{X}, \boldsymbol{y}; \hat{\boldsymbol{a}}) = \frac{1}{2}\sum_i (\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - y_i)^2$ of f is bounded by:*

$$\mathcal{L}_f(\boldsymbol{X}, \boldsymbol{y}; \hat{\boldsymbol{a}}) \leq \frac{1}{2}\left(M \sum_{j=1}^{d} \left[\sigma_j^2 + ((M\eta - 1)\mu_j)^2\right] \right). \tag{1}$$

Note. There is an error in the proof of Theorem 3.1, in [9]. Going from the fourth to the fifth line of Eq. 23, the sum over i on the third term is missing and it should, instead, be $\sum_{ij} \eta^2 M^2 \mu_j^2$. Additionally, the $1/2$ factor does not multiply all terms, when instead it should. We thus provide the correct proof for the theorem with a resulting corrected upper bound:

Proof. Given a training sample $(\boldsymbol{x}_i, y_i)$, with $||\boldsymbol{x}_i|| = 1$, the gradient of the MSE-based loss function $\mathcal{L}$ defined in Eq. 3 w.r.t. $\boldsymbol{a}$ when taking $(\boldsymbol{x}_i, y_i)$ as input is:

$$g(\boldsymbol{x}_i) = \frac{\partial \mathcal{L}(y_i, f(\boldsymbol{x}_i; \boldsymbol{a}))}{\partial \boldsymbol{a}} = \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{a} - y_i \boldsymbol{x}_i \tag{11}$$

We note that:

$$\begin{aligned}(\boldsymbol{a} - g(\boldsymbol{x}_i))^\top \boldsymbol{x}_i - y_i &= \boldsymbol{a}^\top \boldsymbol{x}_i - \boldsymbol{a}^\top \boldsymbol{x}_i \boldsymbol{x}_i^\top \boldsymbol{x}_i + y_i \boldsymbol{x}_i^\top \boldsymbol{x}_i - y_i \\ &= \boldsymbol{a}^\top \boldsymbol{x}_i - (\boldsymbol{a}^\top \boldsymbol{x}_i)(\boldsymbol{x}_i^\top \boldsymbol{x}_i) \\ &= \boldsymbol{a}^\top \boldsymbol{x}_i - \boldsymbol{a}^\top \boldsymbol{x}_i \\ &= 0 \implies y_i = (\boldsymbol{a} - g(\boldsymbol{x}_i))^\top \boldsymbol{x}_i\end{aligned} \tag{12}$$

Then the total training loss among all training samples is given by:

$$\sum_{i=1}^{M} \frac{1}{2}\left(\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - y_i\right)^2 \tag{13}$$

By using Eq. 12, we can rewrite Eq. 13 as follows:

$$\begin{aligned}\sum_{i=1}^{M} \frac{1}{2}\left(\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - y_i\right)^2 &= \sum_{i=1}^{M} \frac{1}{2}\left(\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - (\boldsymbol{a} - g(\boldsymbol{x}_i))^\top \boldsymbol{x}_i\right)^2 \\ &= \sum_{i=1}^{M} \frac{1}{2}\left((\hat{\boldsymbol{a}} - \boldsymbol{a} + g(\boldsymbol{x}_i))^\top \boldsymbol{x}_i\right)^2\end{aligned} \tag{14}$$

Recall the assumption that $\hat{\boldsymbol{a}} = \boldsymbol{a} - \eta \sum_i g(\boldsymbol{x}_i)$; we rewrite Eq. 14 as follows:

$$\sum_{i=1}^{M} \frac{1}{2}\left(\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - y_i\right)^2 = \sum_{i=1}^{M} \frac{1}{2}\left(\left(g(\boldsymbol{x}_i) - \eta \sum_i g(\boldsymbol{x}_i)\right)^\top \boldsymbol{x}_i\right)^2 \tag{15}$$

According to the Cauchy–Schwarz inequality and $\|\mathbf{x}_i\| = 1$, the total training loss is bounded by:

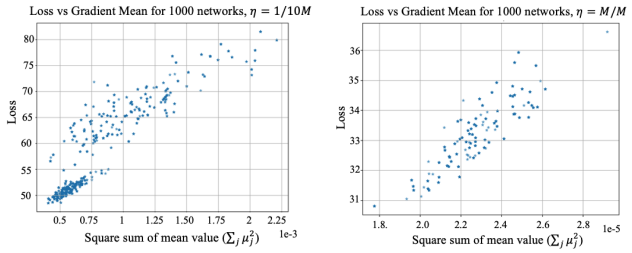
$$\begin{aligned}&\sum_{i=1}^{M} \frac{1}{2}\left(\hat{\boldsymbol{a}}^\top \boldsymbol{x}_i - y_i\right)^2 \leq \\ &\leq \frac{1}{2}\sum_{i=1}^{M} \|(g(\boldsymbol{x}_i) - \eta \sum_i g(\boldsymbol{x}_i))\|^2 \cdot \|\boldsymbol{x}_i\|^2 \\ &= \frac{1}{2}\sum_{i=1}^{M} \|(g(\boldsymbol{x}_i) - \eta \sum_i g(\boldsymbol{x}_i))\|^2 \\ &= \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{d} (g_j(\boldsymbol{x}_i) - \eta M \mu_j)^2 \\ &= \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{d} \left([g_j(\boldsymbol{x}_i)]^2 - 2\eta M \mu_j g_j(\boldsymbol{x}_i) + \eta^2 M^2 \mu_j^2\right) \\ &= \frac{1}{2}\left(\sum_{ij}[g_j(\boldsymbol{x}_i)]^2 + \eta^2 M^3 \sum_j \mu_j^2 - 2\eta \frac{1}{M}\sum_{ij}(M^2 \mu_j g_j(\boldsymbol{x}_i))\right) \\ &= \frac{1}{2}\left(\sum_{ij}[g_j(\boldsymbol{x}_i)]^2 - 2\eta M^2 \sum_j \mu_j^2 + \eta^2 M^3 \sum_j \mu_j^2\right) \\ &= \frac{G}{2} - \frac{\eta}{2} M^2 (2 - \eta M) \sum_j \mu_j^2.\end{aligned} \tag{16}$$

As $G = \sum_j \sum_i [g_j(x_i)]^2 = \sum_j (M\mu_j^2 + M\sigma_j^2)$. Then we can rewrite:

$$\begin{aligned}&\min_{\boldsymbol{a}} \sum_i \mathcal{L}(y_i, f(\boldsymbol{x}_i; \boldsymbol{a})) \leq \\ &\frac{1}{2} M \sum_j \left(\sigma_j^2 + (M^2\eta^2 - 2M\eta + 1)\mu_j^2\right).\end{aligned} \tag{17}$$

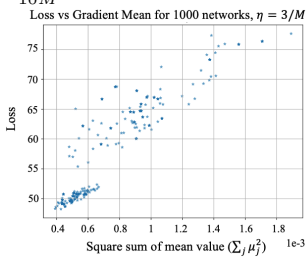
This term is non-negative for all η, therefore it decreseases by decreasing μ_j and σ_j, for any j. Please note that for $\eta = \frac{1}{M}$ our bound reduces to Eq. 6 of ZiCO. □

This result is supported by Fig. 5. Following [9] we built the same experiment setup: we randomly sample 1000 training images from MNIST dataset and normalize them with their L2-norm to create the training set $\mathbb{S}$. With a batch of 1, we train the network for one epoch, compute



(a) $\eta = \frac{1}{10M}$

(b) $\eta = 1$



(c) $\eta = \frac{3}{M}$

Figure 5. Toy example for the positive correlation of μ and the loss $\mathcal{L}$ for 1000 linear networks trained for one epoch on M = 1000 samples with different η.

the gradient w.r.t the network parameters for each individual sample, and update the weights with a learning rate $\eta = \{\frac{1}{10M}; 1; \frac{3}{M}\}$ for three different experiments to provide evidence that our result is valid for a range of η. We compute the square sum of mean gradients (x-axis in the plot) and the total loss (y-axis in the plot). We repeat the process 1000 times on the same $\mathbb{S}$, each time by randomly sampling a different initialization strategy among Kaiming Uniform, Kaiming Normal, Xavier Uniform, and Xavier Normal. The plots show a clear positive correlation for the linear network among the square sum of mean gradients and the loss, as supported by our bound.

B. Overview of the benchmarks

In our experiments, we evaluate the proxies over 14 different tasks and across 6 different search spaces (see Fig. 6). NasBench-101 [17] is a cell-based search space consisting of 423 624 architectures. The design is thought to include ResNet-like and Inception-like DNNs trained multiple times on Cifar-10. In our full evaluation (see C for details) we sampled and ranked 10 000 architectures from this search space. NasBench-201 [4] is a cell-based search space composed of 15 625 architectures (6 466 of which are non-isomorphic) trained on 3 different tasks: Cifar-10, Cifar-100 and ImageNet-16-120. In our full evaluation, we ranked all 15 625 architectures. NasBench-301 [18] (which is not depicted in Fig. 6) is a cell-based search space created as a surrogate NAS benchmark for the DARTS search-space. DARTS is therefore composed of normal and reduction cells for a total of 10^{18} different architectures trained on Cifar-10. In our full evaluation, we ranked 11 221 architectures. TransNAS-Bench-101 [5] is composed of a "Macro" (with 3256 architectures) and a "Micro" (cell-based) (with 4096 architectures) search space. Both Macro and Micro architectures are trained over 7 different tasks taken from the Taskonomy dataset. In our evaluation, we ranked all the 3256 + 4096 architectures. Finally, Autoformer [3] is a one-shot architecture search space for Vision Transformers. We sampled 2000 architectures from the "Small" search-space definition with Embedding dimension (320, 448, 64), $Q - K - V$ dimensions (320, 448, 64), MLP Ratio (3, 4, 0.5), Head Number (5, 7, 1), and Depth Number (12, 14, 1). The tuples of the three values in parentheses represent the lowest, highest, and steps values. We trained the 2 000 architecture on Cifar-10, Cifar-100, ImageNet-1k, SVHN, Pets and Spherical-Cifar100 datasets.

B.1. Autoformer Training

We trained the Autoformer-Small search-space on two A100 Gpus with 80GB of memory each. We followed the standard training procedure introduced in [3] and trained the One-Shot super network on ImageNet-1k splitting the images in 16×16 patches. The training was repeated three times with the weight-entanglement strategy introduced in [3], each time with 500 epochs (with 20 warmup epochs), an AdamW optimizer, 1024 batch size, lr=1e-3, cosine scheduler, weight-decay=5e-2, 0.1 label smoothing and dropout of 0.1. We used the average of the three runs as a test accuracy. The super network has been subsequently fine-tuned on Cifar-10, Cifar-100, Pets, SVHN, and Scifar-100 following the standard DeiT strategy [14]. The 2000 architectures were sampled from the super network after training and directly evaluated with no further fine-tuning on the target dataset.

C. Full search-space

This section extends the experiments from Sec 4.1. For each benchmark and proxy we evaluated the Spearman ρ correlation over a larger collection of architectures, *i.e.* 10 000 fro Nasbench-101, 15 625 for NasBench-201, 11 221 for NasBench-301, 3256 for Tnb101-Macro, 4096 for Tnb101-Micro, and 2000 for Autoformer (see Fig. 7). Most metrics keep stable performance compared to Fig. 3 (that has the results for 1000 architectures), with slightly decreased values for SWAP and ZiCO and a large ρ drop for reg-SWAP which now appears in the first half of the rows. We also present in Fig. 8 a visual comparison between L-SWAG correlation, ZiCO [9], SWAP [11] and the simple metric # parameters for TransNasBench-101 Macro Normal search-space. The plots display the predicted network rankings *vs.* the ground-truth ranking for 1000 architectures. We compare L-SWAG against ZiCO and SWAP as they are the metrics most related to our contribution. We display the results for Macro Normal as it represents a challenging

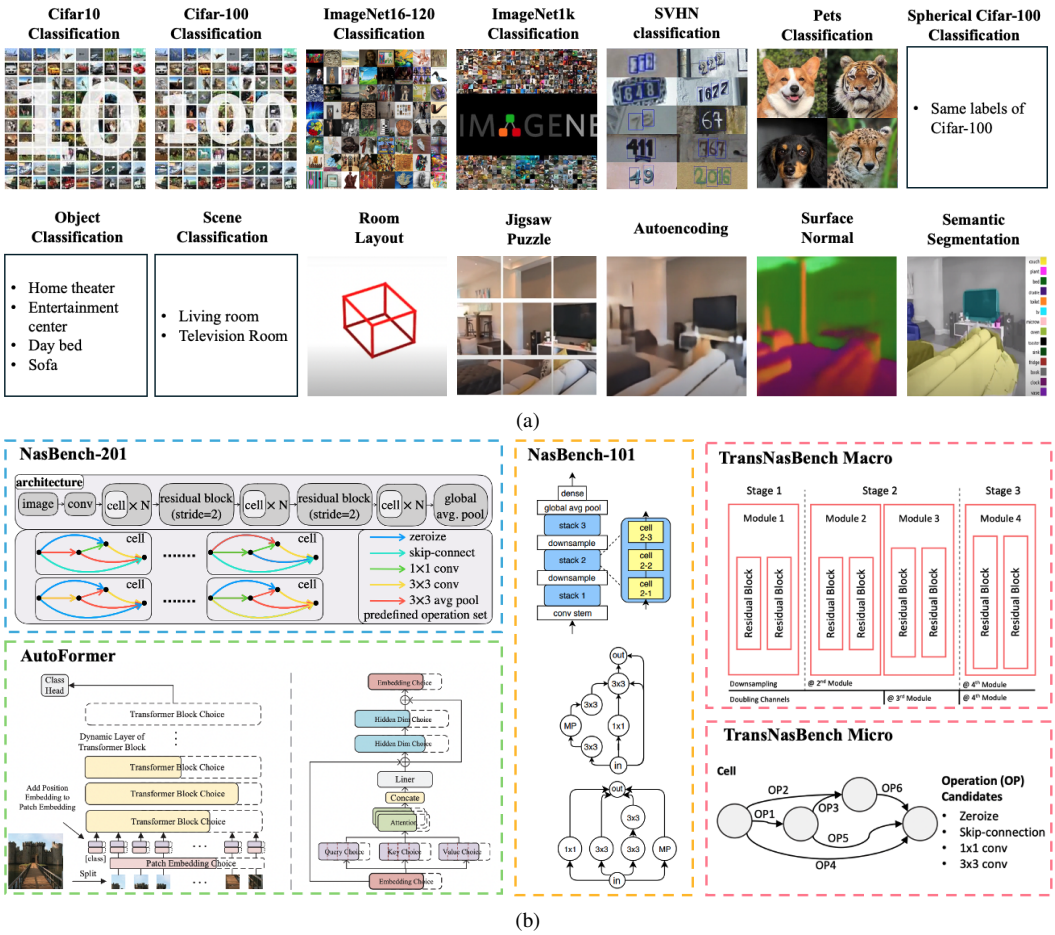


Figure 6. Overview of the deployed datasets (Fig. 6a) and search spaces (Fig. 6b) utilized in our work. We borrow the search-space images from the original NAS benchmark papers [3–5, 17].

benchmark where the benefits of L-SWAG can be better appreciated. Fig. 8a and 8b produce incorrect predictions frequently, leading to low-accuracy networks that are highly ranked and vice-versa. L-SWAG shows the strongest correlation with the ground truth visible through a reduced width across line $y = x$ compared to SWAP.

D. Details from Section 3.1

This section extends the layer-selection choice (Appendix D.1) with the complete set of plots for the gradient statistics behavior introduced in Sec. 3.1, quantitative results on the percentiles ablation depicted in Fig. 2a, and details of the gradient statistics across networks clustered by depth. Appendix D.2 details the choice of direct composing Λ and Ψ in Eq. (1) through multiplication.

D.1. Layer-choice

We organized the plots in Fig. 9 by aggregating search-spaces with similar behavior. These graphs depict the mean and standard deviation of $\frac{1}{\Lambda}$ (introduced in Eq. (1)) across 1000 randomly sampled networks. The goal is to highlight the intensity variation across different percentiles. The analysed search-spaces share different characteristics in the intensity trend, with Fig. 9b displaying NB301 periodic behavior, Fig. 9a highlighting three peaks (percentile 3, 7, and 10) in NB201, Fig. 9c, Fig. 9e and Fig. 9g presenting a unique peak shifted towards the last percentiles, and finally with Fig. 9f and Fig. 9d with an ascending intensity. If we couple these plots with the quantitative results in Tab. 5 which ablates each percentile, and their visual representation in Fig. 2a of the main paper, a clear match between the intensity of $\frac{1}{\Lambda}$ and the Spearman ρ correlation emerges. At this point, one may argue that the influence of the gradient statistics varies depending on the network depth, *i.e.* we cannot average $\frac{1}{\Lambda}$ at the 8^{th} percentile in a network with depth $L = 100$ with $\frac{1}{\Lambda}$ at the 8^{th} percentile in a network with depth $L = 300$. To clear any doubt, we show in Fig. 10 the same results of Fig. 9 obtained by averaging only across networks with a comparable depth. We provide the example for Micro AutoEncoder search space as it represents the trend of all benchmarks. Comparing Fig. 9e with Fig. 10 no substantial differences are observed.

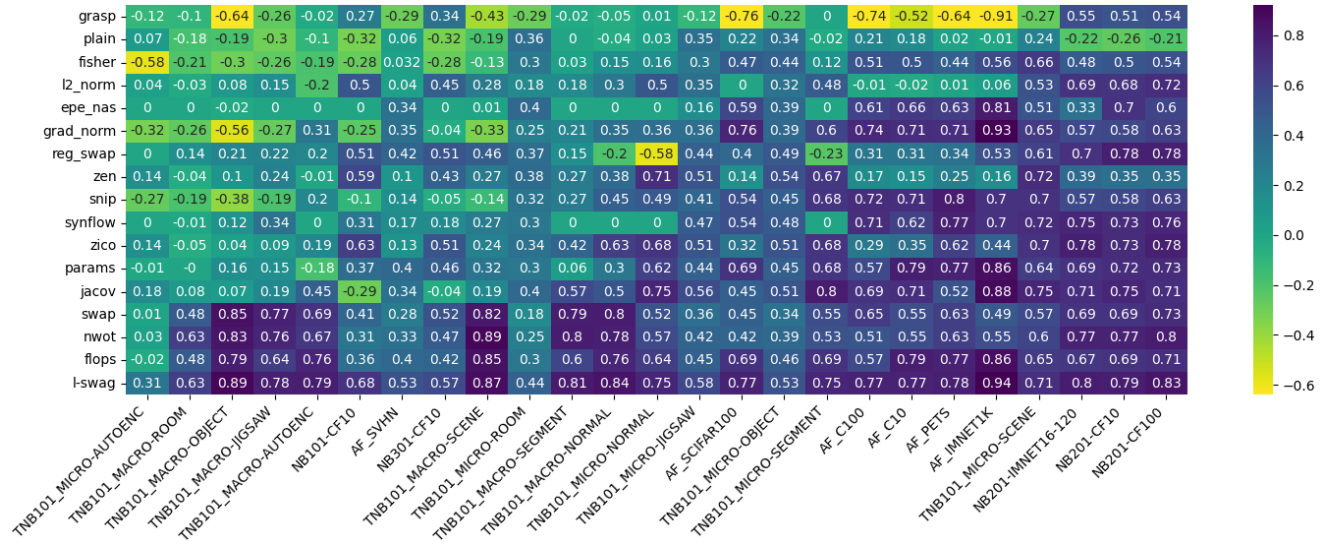
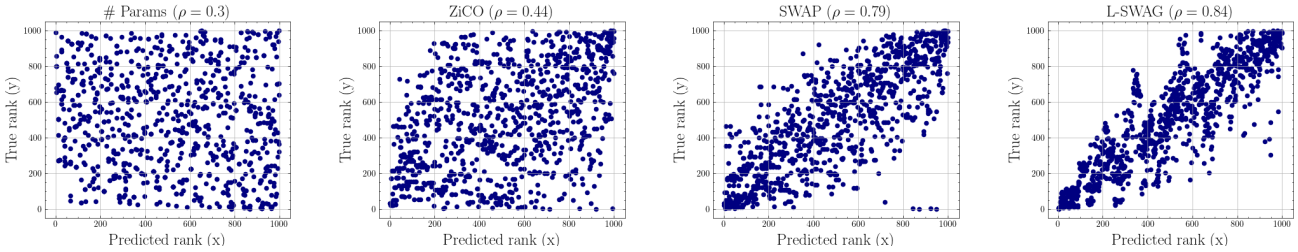


Figure 7. Spearman rank correlation coefficient between ZC proxy values and validation accuracies. Results were obtained from 5 multiple runs. Rows and columns are ordered based on the mean scores. This represents the results of Fig. 3 obtained for a larger number of architectures detailed in Appendix B.



(a) # Params. (b) Zico. (c) SWAP. (d) L-SWAG.

Figure 8. Visual comparison of some ZC-proxy methods in terms of predicted ranking ($x - axis$) and validation accuracy ($y - axis$) on TransNasBench-101 Macro Normal search-space. Each figure reports the Spearman ρ correlation coefficient.

D.2. Multiplication

In Eq. (1) we directly combined Λ and Ψ through multiplication. As different metric combination strategies have been introduced in the literature, in this section we motivate such a choice. [2] combined the ranks of architectures by averaging them across the constituent metrics, a strategy we refer to as "RankAve". The advantage of RankAve lies in its equal weighting of contributions from each metric. However, this method also comes with several limitations. While rank aggregation is viable for certain search spaces and algorithms, it becomes impractical in many scenarios [10]. Additionally, it is an indirect approach and arguably does not create a unified metric but instead offers a way to merge metrics. Similar to the method proposed in [15], we consider addition and multiplication as alternative approaches. Consider two arbitrary metrics, τ_i and τ_j, assumed to be independent random variables, where the samples represent evaluations of a network. For $k \in i, j$, we define $\mu_k = \mathbb{E}[\tau_k]$ and $\sigma_k^2 = \text{Var}(\tau_k)$. Starting with addition, we examine how to combine these metrics such that neither dominates the variance.

$$\text{Var}(\tau_i + \tau_j) = \sigma_i^2 + \sigma_j^2.$$

But what is the effect of the variance on the rankings? Suppose that $\sigma_i \gg \sigma_j$, then [7]:

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

This suggests that the distributional characteristics of $\tau_i + \tau_j$ are primarily influenced by τ_i, resulting in the overall ranking of architectures being controlled by τ_i. Since it is improbable that the variances of the metrics are similar, the metric with the greater variance will dominate. Having excluded addition, we now proceed to evaluate multiplication:

$$\begin{aligned}\text{Var}(\tau_i \cdot \tau_j) = \sigma_i^2\sigma_j^2 + \mu_j^2\sigma_i^2 + \mu_i^2\sigma_j^2 + \mu_i^2\sigma_j^2 \\ + \sigma_i^2\sigma_j^2\left[1 + \left(\frac{\mu_j}{\sigma_j}\right)^2 + \left(\frac{\mu_i}{\sigma_i}\right)^2\right] \qquad (19)\end{aligned}$$

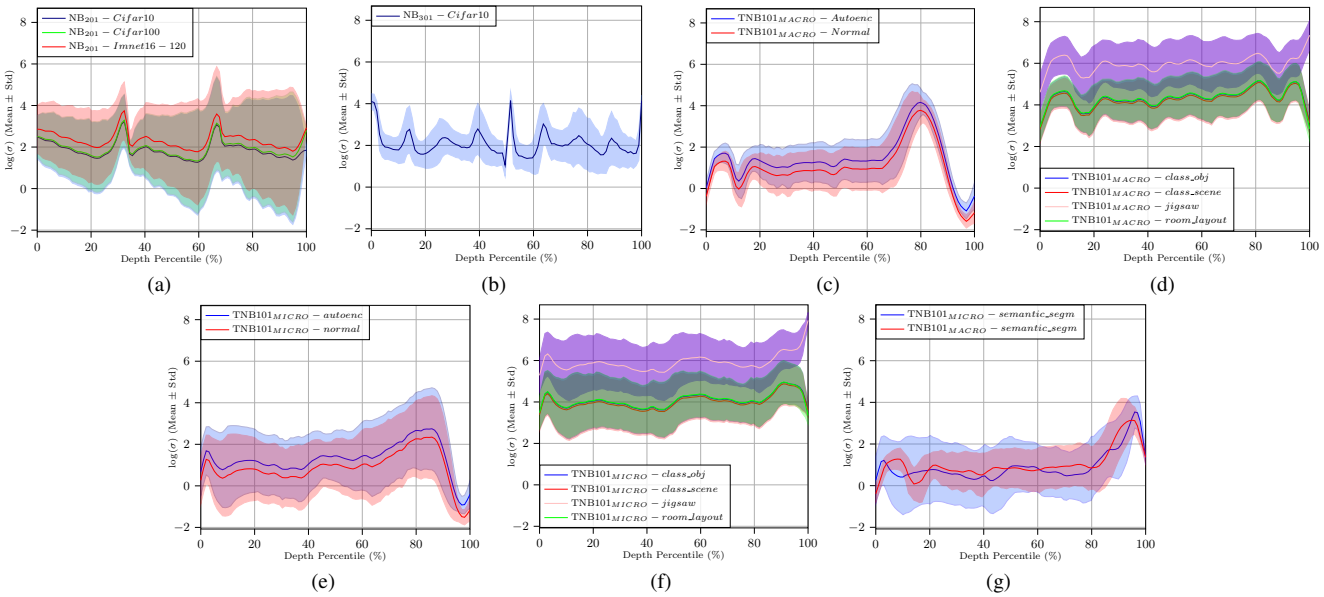


Figure 9. Average gradient statistics across 1000 networks over different depth percentiles. This results completes Fig. 2 in the main paper.

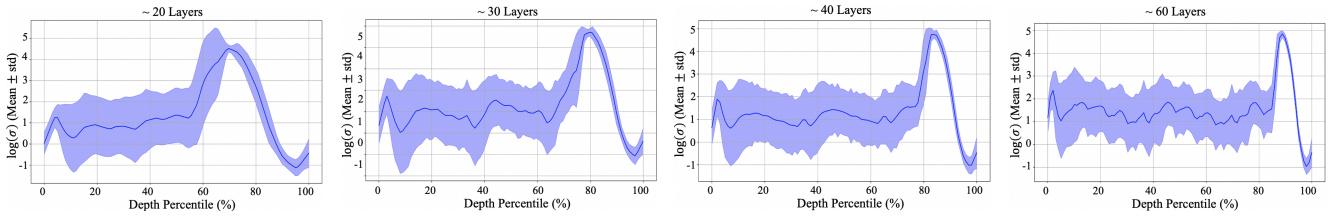
	NB201			NB101	NB301	TNB101-Micro							TNB101-Macro						
Percentile	C10	C100	IN16-120	C10	C10	AE	Room	Obj.	Scene	Jig.	Norm.	Segm.	AE	Room	Obj.	Scene	Jig.	Norm.	Segm.
1	0.720	0.752	0.760	0.665	**0.568**	**0.310**	0.294	0.423	0.596	0.465	0.620	0.510	**0.720**	0.030	**0.890**	0.745	**0.780**	0.700	0.650
2	0.670	0.710	0.723	0.690	0.564	0.200	**0.440**	**0.530**	**0.711**	**0.580**	**0.750**	0.590	0.660	**0.631**	0.700	0.830	0.720	0.800	**0.810**
3	0.711	0.750	0.760	**0.702**	0.565	0.180	0.410	0.490	0.680	0.520	0.740	0.700	0.730	0.630	0.590	0.780	0.680	**0.840**	0.800
4	0.720	0.754	0.763	0.684	0.554	0.190	0.390	0.470	0.670	0.510	0.740	0.704	0.640	0.629	0.620	0.800	0.680	0.740	**0.810**
5	0.690	0.730	0.743	0.687	0.549	0.240	0.420	0.468	0.690	0.540	0.730	0.690	0.580	0.628	0.670	0.810	0.690	0.610	0.740
6	0.720	0.751	0.759	0.680	0.554	0.170	0.390	0.480	0.680	0.520	0.720	0.695	0.620	0.628	0.690	**0.870**	0.710	0.670	0.740
7	0.720	0.751	0.760	0.690	0.550	0.110	0.390	0.470	0.670	0.510	0.630	0.690	0.530	0.625	0.540	0.750	0.730	0.630	0.710
8	0.655	0.690	0.710	0.685	0.540	0.060	0.420	0.490	0.690	0.530	0.530	0.690	0.550	0.625	0.590	0.760	0.630	0.630	0.660
9	0.717	0.749	0.757	0.682	0.540	0.080	0.380	0.460	0.660	0.500	0.510	**0.750**	0.520	0.627	0.600	0.760	0.700	0.550	0.710
10	**0.724**	**0.760**	**0.764**	0.694	0.547	0.000	0.280	0.413	0.640	0.450	0.627	0.520	0.000	0.020	0.630	0.769	0.740	0.000	0.540
ALL	0.710	0.740	0.750	0.651	0.550	0.320	0.330	0.480	0.680	0.520	0.680	0.700	0.700	0.627	0.860	0.780	0.735	0.770	0.780

Table 5. Collection of Spearman's ρ correlation results obtained for the different percentiles. Each row represents an interval, *e.g.* 1 refers to L-SWAG computed with $\hat{l} = 0$ and $\hat{L} = 1$, (meaning that for each row we calculated the metric with two percentiles). "ALL" refers to the metric computed considering all the layers in a network. We highlight in bold the best results.

This highlights that the relationship between the metrics plays a more intricate role in determining the rankings. While not guaranteed, if the metrics' μ_k and σ_k scale proportionally and exhibit similar distributional properties, this approach ensures that neither metric disproportionately dominates the variance. However, a legitimate concern arises: even when using metrics with minimal correlation, the assumption of independence may not always hold. Despite these limitations, we find evidence that, for Λ and Ψ, the contributions of the individual components to the combined scores remain fairly balanced. Although more sophisticated operations than multiplication likely exist for direct composition, this analysis is intended solely as a proof of concept. An additional observation is that directly multiplying the final $\Lambda^{\hat{L}}$ and Ψ results in the loss of much of the layer-wise information that has been gathered. This strengthens the case for our layer-wise multiplication via $\Psi^{\hat{L}}$, effectively performing a dot product of the layer-specific values. Such a layer-wise composition enables an assessment of individual layers based on their specific contribution to the network.

E. Details from Sec. 3.2

This section gives the details for the ZC-proxies z_2, z_3 that were chosen according to LIBRA algorithm and that provided the results of Tab 1. The first ZC-proxy z_1 can be simply derived from Fig. 3 by looking at each column (representing the benchmark) for the highest Spearman's ρ correlation value. The metric that leads to the highest ρ is selected as z_1. The second ZC-proxy z_2 is selected, according to Algorithm 1, by choosing among a filtered set of ZC-proxies z_h. The z_h with the lowest information gain IG between z_1 and z_h becomes z_2. The filtered set is obtained by discarding ZC-proxies with a Spearman's ρ correlation below 0.1 points with respect to z_1 (for all cases otherwise specified). Following this rule, we selected $z_2 = jacov$ for



(a) (b) (c) (d)

Figure 10. Gradient statistics for different networks clustered by depth (20, 30, 40 and 60 layers) in TransBench101-Micro Autoencoder search space.

NB201-C10, $z_2 = zico$ for NB201-C100, $z_2 = nwot$ for NB201-Imnet16-120, $z_2 = swap$ for NB301-C10, $z_2 = jacov$ for TNB101-micro-autoencoder, where the filtered set was obtaining with a tolerance of 0.2 $z_2 = epe\_nas$ for TNB101-micro-room, $z_2 = l\_swag$ for TNB101-micro-object, $z_2 = zen$ for TNB101-micro-scene, $z_2 = zico$ for TNB101-micro-jigsaw, $z_2 = jacov$ for TNB101-micro-normal, $z_2 = l\_swag$ for TNB101-micro-segmentation, $z_2 = nwot$ for TNB101-macro-autoencoder, $z_2 = nwot$ for TNB101-macro-room, where the filtered set was obtaining with a tolerance of 0.2 $z_2 = swap$ for TNB101-macro-object, $z_2 = swap$ for TNB101-macro-scene, $z_2 = swap$ for TNB101-macro-jigsaw, $z_2 = nwot$ for TNB101-macro-normal, $z_2 = nwot$ for TNB101-macro-segmentation. Although the choice in some cases (*e.g.* Macro search-space) was restricted only to two/three ZC-proxies, as most of the z_h had correlation below $\rho = 0.4$, LIBRA could successfully identify the optimal choice. Let us take the example of TNB101-macro-jigsaw: the possible z_h are nwot with $\rho_{nwot} = 0.76$, swap with $\rho_{swap} = 0.74$, and flops with $\rho_{flops} = 0.79$. If we simply chose the metric with the highest ρ (flops) we would obtain a $\rho_{z1,z2} = 0.79$, while LIBRA returns $\rho_{z1,z2} = 0.81$. Finally, in Tab. 6 we present the Pearson's correlation between all ZC-proxies and our chosen bias, *i.e.* the number of parameters. We highlight in bold the ZC-proxy that was chosen according to LIBRA.

F. Influence of the mini-batch size and of random initialization.

We run ablation on the batch size for all measures, including our L-SWAG. We report a representative result for each search-space in Fig. 11. Compared to the other measures in the plots, L-SWAG stabilizes after batch 32 saturating (differently from ZiCo which slightly deteriorates, or to SWAP which in fig. 11a, 11b and 11c has its peak at B=16). We also noticed plain being highly unstable depending on the batch-size. Other metrics (*e.g.* Fisher, # flops *etc.*) with constant values across batches were simply not plotted. We also tested the measure with 3 different random initializations (Xavier, Kaiming and Gaussian) and found the metric to be robust with a std $\sigma = 0.02$.

G. Information theory

For the sake of clarity, we provide full details from Sec. 4.2 and provide the definition of Entropy borrowed from [8]. Given two variables y and z_i, the conditional entropy of y given z_i is defined as:

$$\begin{aligned} H(y|z_i) &= \mathbb{E}[-\log(p(y|z_i))] \\ &= -\sum_{z\in\mathcal{Z}, y\in\mathcal{Y}} p(z,y)\log\frac{p(z,y)}{p(z)} \end{aligned} \tag{20}$$

for two support sets $\mathcal{Y}, \mathcal{Z}$. If we consider entropy as a measure of information—or equivalently, the uncertainty associated with a random variable—conditional entropy reflects the remaining uncertainty after conditioning on another variable. Specifically, $H(y \mid z_i)$ possesses several desirable properties: (1) $H(y \mid z_i) = 0$ if and only if z_i completely determines y; (2) $H(y \mid z_i) = H(y)$ if and only if y and z_i are entirely independent; and (3) $H(y \mid z_{i1}, z_{i2}) = H(y, z_{i1}, z_{i2}) - H(z_{i1}, z_{i2})$. This allows for straightforward computation of conditional entropy when conditioning on multiple random variables. Thus, it serves as an effective metric for quantifying remaining uncertainty or incomplete information. Following the above definition, would require all random variables to be discrete to compute the conditional entropy, which is not our case. Similarly to [8], to properly implement conditional entropy we use Sturge's rule [12] to discretize the float values describing z_is. The heuristic to choose the number of bins is:

$$n_{bins} = \texttt{round(1+3.322*log(N))},$$
with N = sample size.

Information about y does not reveal the exact validation accuracy but rather the interval in which the value falls.

H. LIBRA-NAS and L-SWAG-NAS: more results

We extended the experiments presented in Sec. 4.2 for the Autoformer search space on ImageNet-1k. Rather than comparing with other training-free guided search methods, the focus of this set of experiments is to assess the benefit of ZC-NAS compared to other search methods deployed

	NB201			NB101	NB301	TNB101-Micro							TNB101-Macro						
Name	C10	C100	IN16-120	C10	C10	AE	Room	Obj.	Scene	Jig.	Norm.	Segm.	AE	Room	Obj.	Scene	Jig.	Norm.	Segm.
epe-nas	0.09	0.06	0.09	-0.02	0.07	0.43	0.25	0.22	0.30	0.17	0.40	0.32	0.13	0.12	0.10	0.11	0.02	0.12	0.26
fisher	0.16	0.15	0.07	0.11	0.12	0.16	0.10	0.08	0.18	0.02	0.12	0.10	0.03	0.04	0.02	0.09	0.10	0.16	0.20
flops	0.99	0.99	0.99	1.00	0.98	0.96	0.95	0.99	0.99	1.00	0.98	0.99	0.34	0.49	0.54	**0.53**	0.51	0.39	0.45
grad-norm	0.33	0.40	0.37	0.30	0.55	0.51	0.66	0.70	0.68	0.56	0.47	0.65	0.49	0.34	0.31	0.30	0.20	0.32	0.01
grasp	0.05	0.03	0.13	-0.03	0.16	0.18	0.12	-0.20	-0.23	-0.35	0.20	0.15	0.08	0.16	0.11	0.06	0.08	-0.21	-0.04
l2-norm	0.69	0.69	0.69	0.62	0.99	0.64	0.17	0.79	0.70	0.01	0.64	**0.51**	0.49	0.24	0.76	0.45	0.22	0.85	0.47
jacov	0.06	0.06	0.06	-0.18	0.11	**0.17**	0.00	-0.03	-0.00	**0.41**	0.15	0.18	0.09	0.09	0.07	0.23	0.14	0.32	**0.11**
nwot	0.51	**0.51**	0.50	0.74	0.95	0.42	0.35	0.46	0.40	0.35	0.07	0.35	0.19	**0.24**	**0.31**	0.30	0.21	0.34	0.00
params	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
plain	0.32	0.10	0.23	0.03	0.39	0.12	0.15	0.05	0.08	0.50	0.19	0.10	0.09	0.08	0.17	0.11	0.34	0.06	0.49
snip	0.46	0.45	0.42	0.44	0.55	0.49	0.33	0.22	0.19	0.29	0.55	0.21	0.39	0.28	0.55	0.45	0.49	0.68	0.53
synflow	0.24	0.24	0.24	0.57	0.07	0.41	0.05	0.45	**0.40**	0.44	0.47	0.11	0.27	0.12	0.23	0.21	0.35	0.41	0.23
reg-swap	0.29	0.30	0.21	0.30	**0.44**	-0.06	0.11	-0.09	0.23	-0.78	0.09	-0.15	0.03	-0.09	0.11	-0.02	0.10	0.05	0.03
zico	0.60	0.60	**0.60**	**0.54**	0.97	0.55	**0.48**	**0.54**	0.80	0.44	**0.47**	0.48	0.72	0.59	0.46	0.15	0.41	0.45	0.30
swap	**0.50**	0.51	0.47	0.44	0.50	0.01	0.35	0.30	0.35	0.21	0.35	0.29	**0.32**	0.41	0.54	0.12	**0.11**	**0.39**	0.36
l-swag	0.23	0.24	0.24	0.19	0.32	0.00	0.08	0.15	0.19	0.17	0.15	0.21	0.02	0.16	0.18	0.22	0.10	0.21	0.11
val-acc	0.41	0.55	0.57	0.47	0.52	0.18	0.40	0.53	0.54	0.43	0.44	0.59	0.05	0.07	0.24	0.41	0.16	0.40	0.08

Table 6. Pearson correlation coefficients between predictors and our bias metric (# of Parameters) on different benchmarks. We highlight in bold the value corresponding to the z_3 we chose for LIBRA.

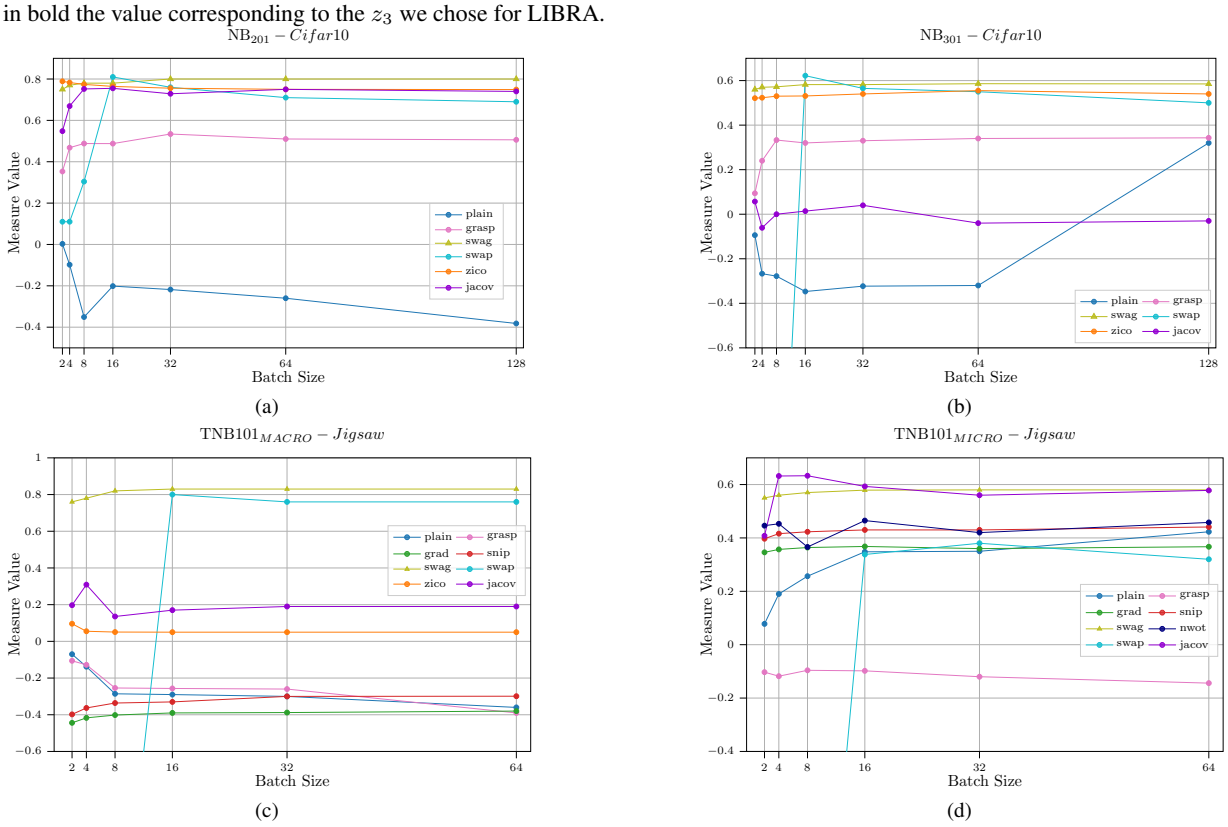


Figure 11. Spearman ρ coefficient consistency of ZC-proxies across different batch sizes.

for the Autoformer search space, including simple random search. Although in Tab. 8 random search still represents a strong alternative, with the best-found architecture after three runs having a test error of 19 %, both L-SWAG NAS and LIBRA-NAS largely improves performance of the found architecture with a negligible search-time. Given the large save in computation time, we hope this set of experiments will further convince the exploration of ZC-proxy design for the ViT search space, to expand research in the video domain. We run additional experiments on BurgerFormer [16] for object detection and instance segmentation on COCO dataset and will add the following results in Sec. H SM. We chose [16] to be comparable with ϵ-GSNR which also validates the metric on these tasks. As ϵ-GSNR, we deployed the found network from BurgerFormer-S space (pre-trained on ImageNet 83.5 % acc.) as the backbone for

	Backbone	AP^b	AP^b_{50}	AP^b_{75}		AP^M	AP^M_{50}	AP^M_{75}
Object Detection	ResNet-50	38.0	58.6	41.4	Instance Segmentation	34.4	55.1	36.7
	PoolFormer	40.1	62.2	43.4		37.0	59.1	36.9
	Swin-T	43.7	66.6	47.7		39.8	63.3	42.7
	ϵ-GSNR	45.0	67.1	49.1		40.7	68.8	43.7
	L-SWAG	**47.5**	**71.4**	**50.3**		**41.4**	**69.7**	**44.2**

Table 7. Comparison with models on COCO dataset.

$\mathcal{B}_{ij}$	Search approach	Params (M)	Search Time (GPU days)	Test Error (%)
AutoFormer Small IMNET1k	Weight entanglement + evolution	22.9	24	18.3
	Random search	23.0	0	19.0
	Classical weight sharing + random†	22.9	-	30.3
	Weight entanglement + random†	22.8	-	18.7
	Classical weight sharing + evolution†	22.9	-	28.5
	ViTAS [13]†	30.5	-	18.0
	NASViT-A0[6]†	[200-300]	-	21.8
	L-SWAG-NAS	23.7	0.05	17.8
	LIBRA-NAS	23.1	0.1	17.0

Table 8. Further comparisons of networks from the Autoformer search space optimized by different NAS methods. While in Tab. 2 we mainly compared the search results obtained running the search algorithm guided by different ZC proxies evaluation, this set of experiments aims instead at showing the benefits of our contributions with respect to other NAS search methods. Random search is performed three times and the best performance is reported. †Results were borrowed directly from [3] and for such a reason no search time is reported, as not available in the original paper.

the Mask R-CNN detector. We used an evolutionary algorithm to search networks within 30M Params.

I. Theoretical intuition behind L-SWAG for ViT

This brief section aims at delivering the intuition behind the design of L-SWAG and the motivation of why it works on ViTs. ViTs use MSA to capture long-range dependencies, but a common issue is rank collapse, where MSA outputs converge to rank-1 matrices, reducing representational diversity. Activation patterns in MSA reflect self-attention's ability to distinguish input tokens. Greater diversity in these patterns at initialization indicates higher expressivity, avoiding rank collapse [1]. While GELU is nonlinear, its smooth transitions still separate input space into "soft regions", which can be counted like in ReLU.Gradient variance ensures trainability, as GELU's smoothness can lead to gradient issues. Together, they provide a holistic measure of both expressivity and trainability.

References

[1] Srinadh Bhojanapalli, Chulhee Yun, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Low-rank bottleneck in multi-head attention models. In *Proceedings of the 37th International Conference on Machine Learning*, pages 864–873. PMLR, 2020. 8

[2] Wei Chen, Xinxin Gong, and Zhiyuan Wang. Neural architecture search on imagenet in four gpu hours: A theoretically inspired perspective. In *International Conference on Learning Representations (ICLR)*, 2021. 4

[3] Xiang Chen, Yiming Wu, Zhiqiang Liu, Ying Wei, Wuyang Zhuang, Shih Yan, Ying Zheng, Zhiqiang Yang, Wenqi Zhang, and Liying Xie. Autoformer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1771–1780, 2021. 2, 3, 8

[4] Ximing Dong and Yiming Yang. Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv preprint arXiv:2001.00326*, 2020. 2

[5] Yawen Duan, Xin Chen, Hang Xu, Zewei Chen, Xiaodan Liang, Tong Zhang, and Zhenguo Li. 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*, pages 5251–5260, 2021. 2, 3

[6] Chengyue Gong, Dilin Wang, Meng Li, Xinlei Chen, Zhicheng Yan, Yuandong Tian, qiang liu, and Vikas Chandra. NASVit: Neural architecture search for efficient vision transformers with gradient conflict aware supernet training. In *International Conference on Learning Representations*, 2022. 8

[7] Kiyosi Itô. *An Introduction to Probability Theory*. Cambridge University Press, Cambridge, 1984. 4

[8] Arjun Krishnakumar, Colin White, Arber Zela, Renbo Tu, Mahmoud Safari, and Frank Hutter. NAS-bench-suite-zero: Accelerating research on zero cost proxies. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 6

[9] Guihong Li, Yuedong Yang, Kartikeya Bhardwaj, and Radu Marculescu. Zico: Zero-shot NAS via inverse coefficient of variation on gradients. In *ICLR*. OpenReview.net, 2023. 1, 2

[10] Min Lin, Peng Wang, Zhiwei Sun, Haoyu Chen, Xiaogang Sun, Qiang Qian, Huchuan 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 (ICCV)*, pages 347–356, 2021. 4

[11] Yameng Peng, Andy Song, Haytham M. Fayek, Vic Ciesielski, and Xiaojun Chang. SWAP-NAS: Sample-wise activation patterns for ultra-fast NAS. In *The Twelfth International Conference on Learning Representations*, 2024. 2

[12] David W. Scott. Sturges' rule. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1(3):303–306, 2009. 6

[13] Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Vision transformer architecture search. In *European Conference on Computer Vision*, 2021. 8

[14] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*, pages 1–8, 2021. 2

[15] Lichuan Xiang, Rosco Hunter, Minghao Xu, Łukasz Dudziak, and Hongkai Wen. Exploiting network compressibility and topology in zero-cost NAS. In *AutoML Conference 2023*, 2023. 4

[16] Longxing Yang, Yu Hu, Shun Lu, Zihao Sun, Jilin Mei, Yinhe Han, and Xiaowei Li. Searching for BurgerFormer with micro-meso-macro space design. In *Proceedings of the 39th International Conference on Machine Learning*, pages 25055–25069. PMLR, 2022. 7

[17] Chris Ying, Aaron Klein, Esteban Real, Eric Christiansen, Kevin P. Murphy, and Frank Hutter. Nas-bench-101: Towards reproducible neural architecture search. In *International Conference on Machine Learning*, 2019. 2, 3

[18] Arber Zela, Julien Siems, Lucas Zimmer, Jovita Lukasik, Margret Keuper, and Frank Hutter. Surrogate nas benchmarks: Going beyond the limited search spaces of tabular nas benchmarks, 2022. 2