PreNAS: Preferred One-Shot Learning Towards Efficient Neural Architecture Search

Haibin Wang [*1] **Ce Ge** [*1] **Hesen Chen** [1] **Xiuyu Sun** [1]

Abstract

The wide application of pre-trained models is driving the trend of once-for-all training in one-shot neural architecture search (NAS). However, training within a huge sample space damages the performance of individual subnets and requires much computation to search for an optimal model. In this paper, we present PreNAS, a search-free NAS approach that accentuates target models in one-shot training. Specifically, the sample space is dramatically reduced in advance by a zero-cost selector, and weight-sharing one-shot training is performed on the preferred architectures to alleviate update conflicts. Extensive experiments have demonstrated that PreNAS consistently outperforms state-of-the-art one-shot NAS competitors for both Vision Transformer and convolutional architectures, and importantly, enables instant specialization with zero search cost. Our code is available at https://github.com/tinyvision/PreNAS.

1. Introduction

Deep learning has made significant strides in a wide range of tasks, including image classification (Tan & Le, 2019; Dosovitskiy et al., 2021), speech recognition (Hsu et al., 2021), and natural language processing (Devlin et al., 2019). One of the key factors contributing to their success is the design of model architectures, which can have a decisive impact on the final performance. However, manual design of model architectures is often time-consuming and requires significant expertise across different tasks. As a result, there is a growing need for automated design process.

Neural architecture search (NAS) is a promising approach to address the challenges, with the goal of finding optimal architectures for a given task with minimal human intervention. One-shot NAS is a kind of NAS family that performs search through a well-trained weight-sharing model, known as a supernet. By training and evaluating possible subnets within a search space using only one single supernet, one-shot NAS significantly reduces the computational cost compared to other NAS methods that train and evaluate every model individually from scratch. One-shot NAS has demonstrated strong performance and has been applied to both convolutional neural networks (CNNs) and Transformer-based architectures (Chitty-Venkata et al., 2022).

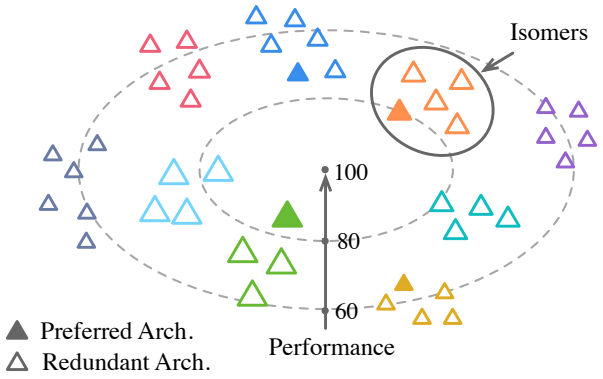


Figure 1. PreNAS first identifies the preferred architectures via zero-cost proxies within similar architectures, denoted as isomers, and then performs concentrated one-shot learning on only preferred architectures to achieve better convergence. The size of triangles implies different resource consumption of the architectures.

In order to reliably search for high-performance models, one-shot NAS solutions mostly employ a *redundant* learning strategy. During the training phase, a plethora of subnets are sampled from a vast search space (can be 10^9 or even much larger) to iteratively update the supernet; during the search phase, sufficient candidate subnets are populated to rank the optimal architecture. While this redundant strategy has indeed achieved good performance and reduced the uncertainty of NAS, it may also be limiting the cutting-edge breakthrough of one-shot NAS. To summarize, there are two specific issues to be concerned. (i) Search efficiency. Each search under the given resource constraints requires the evaluation of several thousand models, which is time-

[*]Equal contribution [1]Alibaba Group, Beijing, China. Correspondence to: Xiuyu Sun <xiuyu.sxy@alibaba-inc.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

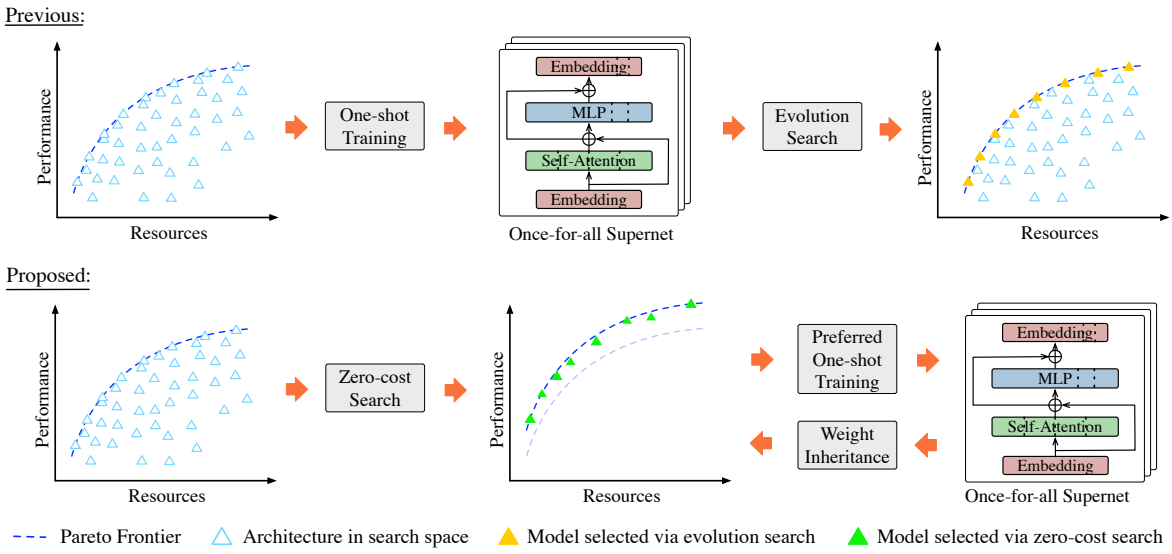


Figure 2. Illustration of PreNAS. Previous one-shot NAS samples all architectures in the search space when one-shot training of the supernet for better evaluation in evolution search. Instead, PreNAS first searches the target architectures via a zero-cost proxy and next applies preferred one-shot training to supernet. PreNAS improves the Pareto Frontier benefited from the preferred one-shot learning and is search-free after training by offering the models with the advance selected architectures from the zero-cost search.

consuming and resource-intensive (Cummings et al., 2022). (ii) Training efficacy. The vast training space makes learnable parameters within the supernet struggle to adapt to various subnets (Liu et al., 2022). The resultant gradient oscillations can impair training convergence and hurt the ultimate performance (Gong et al., 2022).

This work presents a novel *preferred* learning strategy to fulfil the potential of one-shot NAS, namely PreNAS. Considering practical application, NAS methods select high-performance models from the Pareto frontier under given resource constraints, as illustrated in Fig. 1. We are therefore inspired to pre-identify high-quality architectures with various resources in order to enable efficient access to optimal models, achieving nearly search-free specialization. Correspondingly, the selected candidates form a narrow sampling set, which we refer to as the preferred space. During training, the supernet is optimized on the preferred sample space to concentrate the training efficacy, which reduces gradient conflict and thus can potentially reach a better convergence state.

Identifying high-quality architectures without training is a non-trivial task. Another line of NAS research, zero-shot NAS, may provide a feasible solution in this regard. The core of zero-shot NAS lies in designing a zero-cost evaluation metric that serves as a fast proxy with high correlation to the actual performance. In this work, we creatively establish a linkage between one-shot NAS and zero-shot NAS to fully leverage the advantages of both. The key innovation is to enhance scoring-oriented proxy with design capability to handle isomeric Transformers. An undervalued but crucial phenomenon in NAS is that multiple architectures may own the same resource consumption, and this is particularly prevalent in Transformer architectures. The standard Vision Transformers are designed to have homogeneous embedding dimensions across blocks, which produces an interesting property that blocks can be rearranged to form new architectures while maintaining constant parameters and computation. Therefore, the evaluation proxy is revised to own the ability of eliminating isomeric architectures. Fig. 1 provides a schematic diagram for the described process. Overall, the combination of one-shot and zero-shot NAS makes PreNAS not only achieve rapid search of optimal neural architectures, but also obtain well-trained parameters efficiently, enabling out-of-the-box model specialization.

The main contributions are summarized as follows:

- PreNAS, a new learning paradigm that combines one-shot and zero-shot NAS, is proposed to train and search for optimal architectures within a preferred sample space, aiming to improve search efficiency and training efficacy.
- A compound architecture selection mechanism is designed to construct and evaluate high-quality architectures, with a particular focus on addressing structural isomerism in Transformer architectures.
- Extensive experiments and analysis are conducted to verify the effectiveness of PreNAS, showing that it is competent to search for both CNN and ViT architectures with superior performance on various tasks.

2. Background and Related Work

2.1. One-Shot NAS

In one-shot NAS, a weight-sharing supernet is built and trained to jointly optimize subnets within a pre-defined search space. Let $\mathcal{A}$ be the architecture search space and W be all the learnable weights. The objective of one-shot training is to obtain the optimal weights

$$W^* = \arg\min_{W} \mathbb{E}_{\alpha \sim U\{\mathcal{A}\}}[\mathcal{L}_{\text{train}}(\alpha \mid W_\alpha, \mathcal{D}_{\text{train}})], \quad (1)$$

where subnet α is uniformly sampled from the search space $\mathcal{A}$, W_α is the corresponding part of inherited weights, and $\mathcal{L}_{\text{train}}$ is a loss function defined on the training dataset. Upon the well-trained supernet, the best model α^* under the given resource constraint c is searched by ranking subnets as

$$\alpha^* = \arg\max_{\alpha \in \mathcal{A}} \mathcal{P}_{\text{val}}(\alpha \mid W^*_\alpha, \mathcal{D}_{\text{val}}) \ \text{ s.t. } \ r(\alpha) < c, \quad (2)$$

where $\mathcal{P}_{\text{val}}$ evaluates performance on the validation dataset and $r(\cdot)$ reports the resource consumption. During specialization, the search step can be conducted multiple times to obtain deploying models for different resource constraints.

The one-shot NAS (Brock et al., 2018) is proposed to amortize the cost from the training of each model at the beginning. It emphasizes high rank correlation of the accuracy with the quality of subnets (Su et al., 2022), hence requires an additional retraining step after the best architecture is identified (Wu et al., 2019; Li et al., 2020). Recent works propose to alleviate this issue by training a once-for-all supernet in convolution networks (Yu & Huang, 2019; Cai et al., 2020; Yu et al., 2020), Transformer (Wang et al., 2020b; Chen et al., 2021b) and hybrid networks of them (Gong et al., 2022). However, it is difficult to take account of both keeping the performance rank of all subnets in search space and achieving high performances of the target subnets in a once-for-all supernet. FocusFormer (Liu et al., 2022) trains an architecture sampler instead of the uniform sampler in Eq. 1, but does not fully address the problem. Instead, PreNAS isolates the ranking task from the one-shot training via zero-shot NAS in advance.

2.2. Zero-Shot NAS

The advantage of one-shot NAS lies in its ability to simultaneously return both the architectures and optimized parameters. It is more suitable for scenarios where pre-trained models are required, e.g., AutoTinyBERT (Yin et al., 2021) for BERT (Devlin et al., 2019) and LightHuBERT (Wang et al., 2022) for HuBERT (Hsu et al., 2021). However, when only architectures are needed, one-shot NAS would be over cumbersome.

Zero-shot NAS seeks to find high-quality architectures without costly model training. Various zero-cost proxies have been proposed to efficiently estimate architecture quality. The general purpose can be formulized as

$$\alpha^* = \arg\max_{\alpha \in \mathcal{A}} \mathcal{P}_{\text{score}}(\alpha) \quad \text{s.t.} \quad r(\alpha) < c, \quad (3)$$

where $\mathcal{P}_{\text{score}}$ can rapidly score architectures without training and validation on large-scale datasets.

For instance, Mellor et al. (2021) have developed a zero-cost proxy by the overlap of activations between datapoints in untrained networks. Lin et al. (2021) propose Zen-Score to represent the network expressivity. Meanwhile, previous works have leveraged the high correlation of the gradient-based zero-cost proxies with the model performances, such as the gradient norm (Abdelfattah et al., 2021), SNIP (Lee et al., 2019), GraSP (Wang et al., 2020a), SynFlow (Tanaka et al., 2020), NASI (Shu et al., 2022a). Further, Shu et al. (2022b) theoretically prove the connections among different gradient-based zero-cost proxies and propose HNAS to consistently boost existing training-free NAS algorithms. Xu et al. (2021a) apply their proposed gradient-based proxy to RoBERTa (Liu et al., 2019). TF-TAS (Zhou et al., 2022) is the first gradient-based zero-cost proxy especially for ViT (Dosovitskiy et al., 2021). Javaheripi et al. (2022) surprisingly find that the number of decoder parameters in autoregressive Transformers has a high rank correlation with task performance.

Without loss of generality, SNIP (Lee et al., 2019), a representative gradient-based proxy, is adopted in this work. The scoring function of SNIP is defined as:

$$\mathcal{P}_{\text{SNIP}}(\alpha) = \sum_{\theta_i \in \theta_\alpha} \left| \frac{\partial \mathcal{L}}{\partial \theta_i} \odot \theta_i \right|. \quad (4)$$

We use θ to denote randomly initialized parameters to be differentiated from trained weights W. In order to define the loss function $\mathcal{L}$, at least one mini-batch data is required to perform a forward inference. SNIP can be freely replaced with other zero-cost proxies as long as they are competent to select high-quality architectures. As shown in Fig. 2, the models sampled by zero-cost proxy are typically good enough with only a small deviation from the Pareto Frontier.

3. PreNAS

In this section, we begin by revealing the underlying potential of conventional one-shot NAS, and introduce a new preferred learning strategy to fully release its capability as Fig. 2 shows. Then we discuss the isomeric issue in Transformer architectures and detail the adapted zero-cost proxy. Finally, we propose addressing training fairness to balance performance distribution.

3.1. One-Shot NAS with Preferred Learning

As mentioned earlier, conventional one-shot NAS has shortcomings in terms of search efficiency and training efficacy. During the training phase, Eq. 1 expects every subnet $\alpha \in \mathcal{A}$ to be sampled and optimized to reach matchable performance with the architecture quality. However, the search space $\mathcal{A}$ is almost infinitely large, and even redundant training strategy cannot traverse all subnet architectures. More importantly, since the learnable weights W are shared across high-quality and low-quality subnets, the final convergence state is weakened due to conflicting adaptations. Besides, to search for the best architecture from a group of candidate architectures, the performance evaluator $\mathcal{P}_{\text{val}}$ in Eq. 2 could be executed several thousand times in a single search. This search-time overhead is inevitable even with improved strategies like random search (Bender et al., 2018; Li & Talwalkar, 2019), evolution algorithm (Real et al., 2019; Guo et al., 2020) and reinforcement learning (Pham et al., 2018; Tan et al., 2019).

From the perspective of improving search efficiency, the number of candidates to be evaluated should be minimized. A beneficial side effect of this treatment is that it allows for focused updates on fewer subnets, and potentially enables each model to reach a better convergence state. Therefore, we propose performing one-shot learning within a much smaller preferred search space $\widetilde{\mathcal{A}}$ that which satisfies

$$\widetilde{\mathcal{A}} \subset \mathcal{A} \ \text{ and } \ |\widetilde{\mathcal{A}}| \ll |\mathcal{A}|. \tag{5}$$

The reduced search space should be of sufficient quality, so that the best architecture found under the given resource constraint is comparable to the theoretically optimal one in the entire search space, i.e.,

$$\max_{\tilde{\alpha} \in \widetilde{\mathcal{A}}} \mathcal{P}_{\text{val}}(\tilde{\alpha}) \approx \max_{\alpha \in \mathcal{A}} \mathcal{P}_{\text{val}}(\alpha) \tag{6}$$

subject to

$$r(\tilde{\alpha}) < c \ \text{ and } \ r(\alpha) < c. \tag{7}$$

The symbols of weight and dataset in $\mathcal{P}_{\text{val}}$ are omitted for simplicity.

More clearly, the preferred search space is formed by selecting high-quality architectures under various resource constraints:

$$\widetilde{\mathcal{A}} = \left\{ \mathcal{S}(\mathcal{A}_{[c]}, N) \mid \forall c \in \mathcal{C} \right\}. \tag{8}$$

The selector $\mathcal{S}$ chooses the top-N best architectures from the valid subsets and $\mathcal{A}_{[c]}$ is defined as

$$\mathcal{A}_{[c]} = \{ \alpha \mid \alpha \in \mathcal{A} \ \text{ s.t. } \ r(\alpha) < c \}. \tag{9}$$

Constraints $\mathcal{C}$ are a series of limitations on resource consumption, e.g., model size or FLOPs,

$$\mathcal{C} = (a, a + \varepsilon, a + 2\varepsilon, \cdots, b), \tag{10}$$

where ε is the discretization margin and a, b are the lower and upper bounds of the search space.

Upon the preferred search space, the supernet is trained in a weight-entanglement manner (Chen et al., 2021b) but focusing on only high-quality architectures. The preferred one-shot training process can be rewritten as:

$$\widetilde{W}^* = \arg\min_{\widetilde{W}} \mathbb{E}_{\alpha \sim B\{\widetilde{\mathcal{A}}\}} [\mathcal{L}_{\text{train}}(\alpha \mid \widetilde{W}_\alpha, \mathcal{D}_{\text{train}})]. \tag{11}$$

Concentrated optimization on high-quality subnets avoids sharing weights with poor architectures and results in superior individual performance, which is confirmed by our state-of-the-art results reported in Section 4.2.

In the specialization stage, the optimal architectures under given resource constraints can be directly obtained:

$$\widetilde{\mathcal{A}}^* = \widetilde{\mathcal{A}}. \tag{12}$$

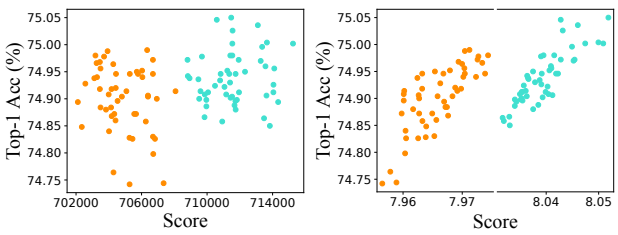
This procedure does not require costly search and evaluation on a large number of candidates. It achieves zero-cost specialization in most cases and is sufficient for general uses. For rare finer-grained use cases, there is only a small additional overhead to rank a few subnets.

3.2. Zero-Cost Transformer Selector

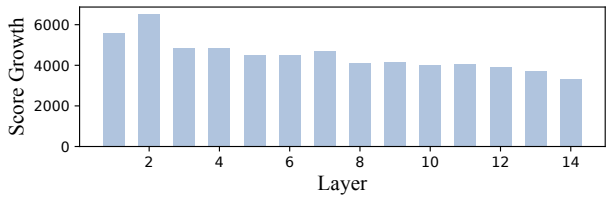
Proxy Confusion The foundation of PreNAS lies in selecting high-quality architectures before data-based training, i.e., the selector $\mathcal{S}$ in Eq. 8. This is a non-trivial task and we resort to the proxy techniques in zero-shot NAS research. A naive idea would be to use gradient-based proxies such as SNIP (Lee et al., 2019) to search for qualified subnets. After preliminary experiments, we found that SNIP as well as other similar proxies perform well across architectures of varying capacity. The computed scores have a high correlation to true accuracy. However, it encounters confusion in distinguishing architectures with the same resource consumption.

Definition 3.1 (Transformer isomers)**.** For a L-block Transformer architecture $\alpha \coloneqq (\beta_1, \beta_2, \ldots, \beta_L)$ with β_i the configurations of i-th block, all architectures produced by reordering α are isomers.

Understanding Isomers We attribute the confusion phenomenon to the isomerism of Transformer architectures. SNIP was originally proposed for CNN architectures and shows a high correlation with model size. However, Transformer isomers have constant model sizes and FLOPs, which deceives the SNIP score. Fig. 3(a) illustrates two groups of isomers, and it is clear that there is no significant correlation between SNIP and accuracy. By inspecting per-block scores of the supernet, we observe from Fig. 3(c) that the unit contribution of lower blocks is relatively higher.



(a) SNIP (b) Normalized SNIP



(c) Score sensitivity

Figure 3. (a) The SNIP scores are uncorrelated with the test accuracy of various subnets in two isomers. (b) The SNIP scores with the layer normalization have significant positively correlation with the test accuracy of various subnets in the two isomers. (c) The growth of the SNIP score in each layer with the increase of the MLP ratio from 3.5 to 4.

Therefore, SNIP tends to select architectures with wider lower blocks, which typically exhibit mediocre performance. Refer to Appendix A for more detailed analyses. It is noteworthy that the slight advantage of lower blocks can be easily dominated by scaling embedding width and the number of blocks, which explains why the SNIP still works under different resource consumption.

Normalized Isomer Proxy Based on the above analysis, we propose a natural solution that normalizes SNIP on a per-layer basis. The revised scoring function of normalized SNIP is defined as:

$$\mathcal{P}^{\text{norm}}_{\text{SNIP}} = \sum_{l=1}^{L} \left(\sum_{\theta^l_\alpha} \left| \frac{\partial \mathcal{L}}{\partial \theta^l_\alpha} \odot \theta^l_\alpha \right| \Big/ \sum_{\theta^l} \left| \frac{\partial \mathcal{L}}{\partial \theta^l} \odot \theta^l \right| \right), \tag{13}$$

where θ^l denotes the randomly initialized parameters of the supernet at l-th block and θ^l_α is the inherited parameters of subnet α. As Fig. 3(b) shows, the normalized SNIP score shows a positive correlation with accuracy for isomers. It is noteworthy that each isomer group may contain a large number of architectures, and applying $\mathcal{P}^{\text{norm}}_{\text{SNIP}}$ on all of them is inefficient. Other approximation techniques like random search or evolutionary algorithm can help speed up but do not guarantee always getting the best results. Therefore, we provide an efficient allocation algorithm based on $\mathcal{P}^{\text{norm}}_{\text{SNIP}}$ that greedily constructs the top-scoring isomer in $\mathcal{O}(1)$ complexity. The algorithm is detailed in Appendix B.

Table 1. The search space of PreNAS for Vision Transformer. Following AutoFormer (Chen et al., 2021b), three supernets are built spanning different parameter scales. The triplet of each variable factor means the lower bound, upper bound, and incremental step, respectively. The Q-K-V dimensions, numbers of heads, and MLP ratios varies across different layers.

	Supernet-Tiny	Supernet-Small	Supernet-Base
Embed Dim	(192, 240, 24)	(320, 448, 64)	(528, 624, 48)
Q-K-V Dim	(192, 256, 64)	(320, 448, 64)	(512, 640, 64)
MLP Ratio	(3.5, 4, 0.5)	(3, 4, 0.5)	(3, 4, 0.5)
Head Num	(3, 4, 1)	(5, 7, 1)	(8, 10, 1)
Depth Num	(12, 14, 1)	(12, 14, 1)	(14, 16, 1)
Params Range	5.4 – 10.5M	13 – 34M	42 – 76M

Finally, the zero-cost selector $\mathcal{S}$ in Eq. 8 is a compound two-step operation. It first apply $\mathcal{P}^{\text{norm}}_{\text{SNIP}}$ on isomers to elect representatives and then apply $\mathcal{P}_{\text{SNIP}}$ to select the top-N preferred architectures under the given resource constraint.

3.3. Performance Balancing

We discuss training fairness and demonstrate how to distribute learning performance more fairly across subnets. As indicated in Eq. 1, the usual one-shot training samples subnets uniformly from the vast search space, i.e., $\alpha \sim U\{\mathcal{A}\}$. In our preferred search space $\widetilde{\mathcal{A}}$, the candidate architectures are evenly distributed in terms of resource consumption. However, a uniform sampling on $\widetilde{\mathcal{A}}$ do not result in fair updates in terms of subnets. The variable building factors like embedding dimension, number of blocks, and number of attention heads do not appear equally, among which the former two have a greater impact. Hence, we propose grouping the architectures in $\widetilde{\mathcal{A}}$ by both their embedding dimensions and depths and perform uniform sampling on this calibrated distribution, i.e., $\alpha \sim B\{\widetilde{\mathcal{A}}\}$ in Eq. 11.

4. Experiments

4.1. Setup

Search Space We employ the same search space of AutoFormer (Chen et al., 2021b) including five variable factors in Transformer blocks: embedding dimension, Q-K-V dimension, number of heads, MLP ratio, and network depth, as shown in Tab. 1. We also apply the paradigm of PreNAS to CNNs with the same search space of BigNAS (Yu et al., 2020) in Tab. 2.

Implementation Details We implemented PreNAS upon the PyTorch (Paszke et al., 2019) framework with improvements from the timm (Wightman, 2019) library. The supernets are trained following the recipe outlined in AutoFormer (Chen et al., 2021b), where multiple data augmenta-

Table 2. The search space of PreNAS based on MobileNetV2. Following BigNAS (Yu et al., 2020), the building blocks include vanilla convolutions and inverted bottleneck residual blocks (MBConv) (Sandler et al., 2018).

Stage	Operator	Resolution	#Channels	#Layers	Kernel
	Conv	192 – 320	32 – 40	1	3
1	MBConv1	96 – 160	16 – 24	1 – 2	3
2	MBCouv6	96 – 160	24 – 32	2 – 3	3
3	MBCouv6	48 – 80	40 – 48	2 – 3	3, 5
4	MBCouv6	24 – 40	80 – 88	2 – 4	3, 5
5	MBCouv6	12 – 20	112 – 128	2 – 6	3, 5
6	MBCouv6	12 – 20	192 – 216	2 – 6	3, 5
7	MBCouv6	6 – 10	320 – 352	1 – 2	3, 5
	Conv	6 – 10	1280 – 1408	1	1

tion and regularization techniques are utilized to facilitate convergence, including RandAugment (Cubuk et al., 2020), mixup (Zhang et al., 2018), CutMix (Yun et al., 2019), Random Erasing (Zhong et al., 2020), stochastic depth (Huang et al., 2016), Repeated Augmentation (Berman et al., 2019; Hoffer et al., 2020) and Label Smoothing (Szegedy et al., 2016; Yuan et al., 2020). The detailed hyper-parameters are presented in Appendix C. The input images are all resized to 224×244 and split into patches of size 16×16. We use the AdamW optimizer with a mini-batch size of 1024. The learning rate is initially set to 1e-3 and decays to 2e-5 through a cosine scheduler in 500 epoches. The discretization margin ε is set to 1M. We conducted experiments and measured design time on NVIDIA A100 GPUs.

4.2. Main Results

Comparison with NAS ViT The results of PreNAS are presented in Tab. 3. In accordance with DeiT (Touvron et al., 2021), we compare the best results on three specifications, viz., PreNAS-Tiny, PreNAS-Small, and PreNAS-Base, each corresponding to a parameter limit of 6M, 23M, and 54M, respectively. The recent AutoFormer (Chen et al., 2021b) is the most relevant NAS competitor, which adopted a redundant one-shot training strategy and searched for optimal subnets by evolutionary algorithm. It can be observed that our PreNAS surpasses AutoFormer on both top-1 and top-5 accuracy under all three resource constraints. Besides, it is important that these achievements are obtained with highly competitive search efficiency. Specifically, AutoFormer utilizes evolutionary algorithm and requires at least 90 GPU hours to safely provide an optimal model. While PreNAS is indeed search-free after one-shot training and can instantly provide a desired model under the given resource constraint.

Comparison with Handcrafted ViT Some competitive ViT models that were elaborately designed by humans are also included, among which DeiT (Touvron et al., 2021), ConViT (d'Ascoli et al., 2021), TNT (Han et al., 2021) and

Table 3. Comparison of different Vision Transformers on ImageNet. ‡Hybrid models of convolutions and Transformer blocks.

Model	Top-1 (%)	Top-5 (%)	#Params (M)	FLOPs (G)
LVT‡	74.8	92.6	5.5	0.9
DeiT-Ti	72.2	91.1	5.7	1.2
ConViT-Ti	73.1	91.7	6.0	1.0
TNT-Ti	73.9	91.9	6.1	1.4
AutoFormer-Ti	74.7	92.6	5.9	1.3
PreNAS-Ti	**77.1**	**93.4**	**5.9**	**1.4**
BoTNet-S1-59‡	81.7	95.8	33.5	7.3
DeiT-S	79.8	95.0	22.1	4.7
ConViT-S	81.3	95.7	27.0	5.4
TNT-S	81.5	95.7	23.8	5.2
T2T-ViT-14	81.7	-	21.5	6.1
AutoFormer-S	81.7	95.7	22.9	4.9
PreNAS-S	**81.8**	**95.9**	**22.9**	**5.1**
BoTNet-S1-110‡	82.8	96.3	55	11
DeiT-B	81.8	95.6	86	18
ConViT-B	82.4	95.9	86	17
AutoFormer-B	82.4	95.7	54	11
PreNAS-B	**82.6**	**96.0**	**54**	**11**

T2T-ViT Yuan et al. (2021) are pure ViT architectures, while LVT (Yang et al., 2022) and BoTNet (Srinivas et al., 2021) are hybrid architectures built of convolutions and Transformer blocks. While boosting accuracy, PreNAS also tend to have fewer parameters and FLOPs than manual designs, especially in larger-scale models. For example, PreNAS-Base improves top-1 accuracy from 81.8% to 82.6% with only 63% parameters and 61% FLOPs of DeiT-B. It is noteworthy that hybrid architectures utilize convolutions to effectively down-sample features, which yields significant advantageous in computation reduction. Surprisingly, PreNAS demonstrates comparable performance to BoTNet with similar or even fewer parameters and FLOPs.

In summary, the experimental results firmly reveal the effectiveness of PreNAS. It not only outperforms pure state-of-the-art Vision Transformers, but also reaches an extraordinary level of parity with compact hybrid architectures.

4.3. Analysis and Ablation study

Quality of Preferred Architectures In PreNAS, the quality of preferred architectures $\widetilde{\mathcal{A}}$ is crucial and is the basis of the entire framework. Therefore, we take the architectures selected by zero-cost selector $\mathcal{S}$ (without subsequent one-shot training) and train them from scratch to solely verify its effectiveness. The results are shown in Tab. 4 and all the methods are fairly trained for 300 epochs with the same recipe. In both the Tiny and Small search spaces, the trained accuracy of PreNAS is superior to other competitors, ex-

Table 4. Performance comparison of subnets trained from scratch (300 epochs and 1024 batch size). SNIP is applied by replacing the normalized SNIP in the first step of seletor $\mathcal{S}$. The subscript $_{ZS}$ means using only the zero-shot selector of PreNAS.

METHOD	TOP-1 (%)	TOP-5 (%)	#PARAMS (M)	TIME (HOURS)
SNIP-TI	74.4	92.4	6.0	0.1
AUTOFORMER-TI	74.4	92.4	5.9	415
TF-TAS-TI	75.2	92.7	6.2	12
PRENAS-TI $_{ZS}$	**74.8**	**92.6**	**6.0**	**0.1**
SNIP-S	81.2	95.7	23	0.3
AUTOFORMER-S	81.5	95.6	23	667
TF-TAS-S	81.4	95.7	24	17
PRENAS-S $_{ZS}$	**81.5**	**95.8**	**23**	**0.3**

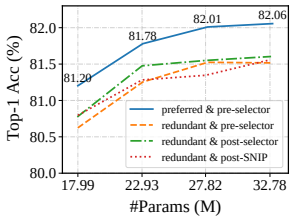


Figure 4. ImageNet accuracy of different routines of zero-cost proxies and one-shot training in the Small space.

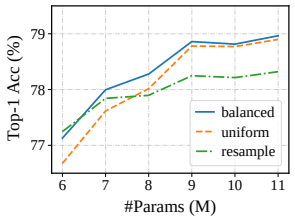
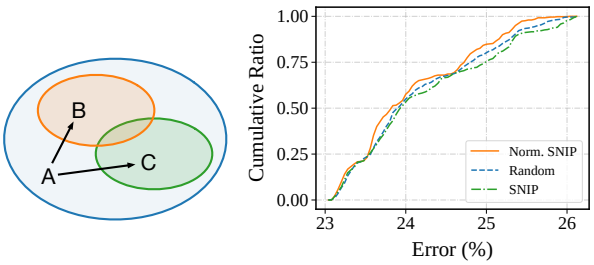


Figure 5. ImageNet accuracy of different sampling strategies during one-shot training in the Tiny space.

cept for TF-TAS-Ti, which actually has advantageous more parameters. We attribute the benefits of PreNAS to its normalization of isomer proxy, which eliminates the preference for wider lower blocks in SNIP. For example, the MLP ratios from low level layer to high level layer in SNIP-Ti are 4.0, 4.0, 4.0, 4.0, 4.0, 4.0, 4.0, 4.0, 4.0, 3.5, 3.5, 3.5 respectively, while they are 4.0, 4.0, 4.0, 4.0, 4.0, 3.5, 3.5, 4.0, 4.0, 3.5, 4.0, 4.0 in PreNAS-Ti. From the perspective of efficiency, our two-step selector is competent as an excellent zero-shot proxy. The search time is nearly zero at the same level as SNIP, and the architecture quality is comparable or even better than evolutionary search in one-shot AutoFormer. TF-TAS (Zhou et al., 2022) is a recently proposed zero-shot proxy specifically for Transformers. Although it promoted the search efficiency by $48\times$ compared to AutoFormer, it still consumes at least 12 hours to offer an architecture via massive forward inferences. In contrast, PreNAS requires only one forward and backward propagation via a mini batch and all subnets directly inherit the initialized weights and gradients from supernet.

Investigation of Preferred Training PreNAS is a combination of zero-shot selector and preferred one-shot training. It is possible to combine the proposed selector with conventional redundant one-shot training. Fig. 4 shows



(a) Search spaces (b) Error distributions

Figure 6. Model quality in different search spaces. (a) Space A is the initial entire space, B is the refined search space by applying layer-normalized SNIP to isomers as in Eq. 13, and C is selected by original SNIP without normalization. (b) Error rate statistic of ImageNet classification in different spaces. We randomly sampled 500 models in each space to inspect their error distributions under redundant one-shot training. Cumulative ratio represents the percentage of sampled models with error below a certain value.

several possible alternatives. "preferred & pre-selector" is the recipe of PreNAS. "redundant & pre-selector" means training supernet in the redundant manner and selecting subnets beforehand with randomly initialized parameters, while "redundant & post-selector" means that subnets are selected afterwards from supernet with well-trained parameters. The figure confirms the effectiveness of preferred training in convergence. Specifically, with the same selected subnets in "preferred & pre-selector" and "redundant & pre-selector", the preferred training improves accuracy by about 0.5% for each subnet comparing to the redundant ones. By concentrating on high-quality architectures, the preferred training encounters less update conflicts and thus achieves more adequate optimization for all selected subnets.

Effect of Normalized Proxy We further analyze the importance of normalized isomer proxy in our twp-step selector $\mathcal{S}$. In the first step, we preserve a unique representative from each group of Transformer isomers using a normalized SNIP score $\mathcal{P}^{\text{norm}}_{\text{SNIP}}$, which forms a refined search space B as in Fig. 6(a). As a comparison, we apply the original SNIP without layer normalization in first step to obtain the search space C. We follow the criteria of RegNet (Radosavovic et al., 2020) to characterize the qualities of refined search spaces. We randomly sample 500 architectures from the entire search space A to analyze performance distribution. The corresponding isomers in B and C are picked to keep the constant parameters for fair comparison. As shown in Fig. 6(b), the curves imply that the quality of B covers A and C and the models in C perform worst.

Effect of Performance Balancing During one-shot training, the optimizer will update the learnable weights of a subnet chosen from the search space at each iteration. A

Table 5. Transfer learning on different downstream tasks with ImageNet pre-training.

MODEL	#PARAMS	CIFAR-10	CIFAR-100	FLOWERS	CARS	PETS	INAT-19
EFFICIENTNET-B5	30M	98.7	91.1	98.5	-	-	-
GRAFIT RESNET-50	25M	-	-	98.2	92.5	-	75.9
VIT-B/16	86M	98.1	87.1	89.5	-	93.8	-
VIT-L/16	307M	97.9	86.4	89.7	-	-	-
DEIT-B	86M	99.1	90.8	98.4	92.1	-	77.7
VITAE-S	24M	98.8	90.8	97.8	91.4	94.2	76.0
DEARKD-S	22M	98.4	89.3	97.4	91.3	-	-
PRENAS-S	**23M**	**99.1**	**91.2**	**97.6**	**92.2**	**94.9**	**76.4**

Table 6. Comparative results of CNN architectures on ImageNet. The paradigm of PreNAS is applied to BigNAS search space.

GROUP	MODEL	TOP-1 (%)	FLOPS (M)
200M	BIGNAS-S	76.5	242
	PRENAS-S $_{CNN}$	**77.4**	**237**
400M	BIGNAS-M	78.9	418
	PRENAS-M $_{CNN}$	**79.9**	**413**
600M	BIGNAS-L	79.5	586
	PRENAS-L $_{CNN}$	**80.0**	**528**
1000M	BIGNAS-XL	80.9	1040
	PRENAS-XL $_{CNN}$	**81.4**	**986**

basic sampling method is to uniformly choose candidate architectures from the preferred space $\widetilde{\mathcal{A}}$ with equal probability. However, this can lead to a suboptimal optimization of certain weights as the variable building factors are trained to different frequencies. As shown in Fig. 5, uniform sampling leads to poor performance of small models due to the uncommon selection of variable factors. As a contrast, we also experimented with the resampling method employed in redundant one-shot training, i.e., randomly construct new architectures from the decomposed search space using variable factors as the minimal component. This helps to improve small models, but the performances of most subnets deteriorate due to the growing number of subnets for optimization. The balanced sampling ensures that the most critical embedding dimension and depth are fairly updated, thus achieving consistently superior performance.

4.4. Transfer Learning Results

As the power of generalization is important for deep learning models, we further evaluate PreNAS on various downstream tasks to measure its transfer learning capability. Following convention, the compared models are pre-trained on ImageNet and then fine-tuned on the target datasets. As shown in Tab. 5, we present results on CIFAR-10/100 (Krizhevsky & Hinton, 2009), Flowers-102 (Nilsback & Zisserman, 2008), Stanford Cars (Krause et al., 2013), Oxford-IIIT Pets (Parkhi et al., 2012), and iNaturalist 2019 (Horn et al., 2018). Our small model is able to outperform ViT and DeiT on multiple datasets with several-fold fewer parameters. PreNAS-S is mostly superior to similar-scale ViTAE-S (Xu et al., 2021b) and DearKD-S (Chen et al., 2022) and is on par with convnet models.

4.5. CNN Results

Although the main focus of PreNAS is on Vision Transformer, the concept of preferred learning for one-shot NAS is general and can be applied to various architectures. Here we demonstrate the adaptation on CNN architectures to prove its versatility and extensibility. We choose BigNAS (Yu et al., 2020) as the experimental basis, which is a popular single-stage NAS framework for CNN architectures. The paradigm of PreNAS can be easily migrated to BigNAS, with the main effort being replacing search space of transformer blocks with convolutional layers. The experimented search space is detailed in Tab. 2. Since isomers rarely appear in CNN architectures due to the down-sampling nature, the normalization step of Eq. 13 can be skipped. The benchmark results are shown in Tab. 6. Our PreNAS achieves from 0.5% to 1.0% better accuracy for different model sizes than BigNAS in terms of similar FLOPs.

5. Conclusion

In this paper, we proposed PreNAS, a one-shot neural architecture search method that reduces the training space by identifying promising isomers and further pruning the training space through sparsification. By doing so, PreNAS is able to conduct one-shot learning on a more focused set of architectures, leading to higher accuracy and is search-free after training by the zero-cost search in advance. Our experiments demonstrate that PreNAS is able to produce

highly accurate models with significantly reduced search times compared to other NAS methods. We believe that PreNAS represents a promising approach to improving the efficiency and effectiveness of NAS, and we look forward to further exploring its potential in future research.

References

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

Bender, G., Kindermans, P., Zoph, B., Vasudevan, V., and Le, Q. V. Understanding and simplifying one-shot architecture search. In *International Conference on Machine Learning (ICML)*, volume 80, pp. 549–558, 2018.

Berman, M., Jégou, H., Vedaldi, A., Kokkinos, I., and Douze, M. MultiGrain: A unified image embedding for classes and instances. *arXiv preprint arXiv:1902.05509*, 2019.

Brock, A., Lim, T., Ritchie, J. M., and Weston, N. SMASH: One-shot model architecture search through hypernetworks. In *International Conference on Learning Representations (ICLR)*, 2018.

Cai, H., Gan, C., Wang, T., Zhang, Z., and Han, S. Once-for-All: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations (ICLR)*, 2020.

Chen, H., Lin, M., Sun, X., and Li, H. NAS-Bench-Zero: A large scale dataset for understanding zero-shot neural architecture search. https://openreview.net/forum?id=hP-SILoczR, 2021a.

Chen, M., Peng, H., Fu, J., and Ling, H. AutoFormer: Searching transformers for visual recognition. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12250–12260, 2021b.

Chen, X., Cao, Q., Zhong, Y., Zhang, J., Gao, S., and Tao, D. DearKD: Data-efficient early knowledge distillation for vision transformers. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12042–12052, 2022.

Chitty-Venkata, K. T., Emani, M., Vishwanath, V., and Somani, A. K. Neural architecture search for transformers: A survey. *IEEE Access*, 10:108374–108412, 2022.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, pp. 3008–3017, 2020.

Cummings, D., Sarah, A., Sridhar, S. N., Szankin, M., Munoz, J. P., and Sundaresan, S. A hardware-aware framework for accelerating neural architecture search across modalities. *arXiv preprint arXiv:2205.10358*, 2022.

d'Ascoli, S., Touvron, H., Leavitt, M. L., Morcos, A. S., Biroli, G., and Sagun, L. ConViT: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 2286–2296, 2021.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–4186, 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Gong, C., Wang, D., Li, M., Chen, X., Yan, Z., Tian, Y., Chandra, V., et al. NASViT: Neural architecture search for efficient vision transformers with gradient conflict aware supernet training. In *International Conference on Learning Representations (ICLR)*, 2022.

Guo, Z., Zhang, X., Mu, H., Heng, W., Liu, Z., Wei, Y., and Sun, J. Single path one-shot neural architecture search with uniform sampling. In *European Conference on Computer Vision (ECCV)*, volume 12361, pp. 544–560, 2020.

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., and Wang, Y. Transformer in transformer. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 15908–15919, 2021.

Hoffer, E., Ben-Nun, T., Hubara, I., Giladi, N., Hoefler, T., and Soudry, D. Augment your batch: Improving generalization through instance repetition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8126–8135, 2020.

Horn, G. V., Aodha, O. M., Song, Y., Cui, Y., Sun, C., Shepard, A., Adam, H., Perona, P., and Belongie, S. J. The inaturalist species classification and detection dataset. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8769–8778, 2018.

Hsu, W., Bolte, B., Tsai, Y. H., Lakhotia, K., Salakhutdinov, R., and Mohamed, A. HuBERT: Self-supervised speech

representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460, 2021.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *European Conference on Computer Vision (ECCV)*, volume 9908, pp. 646–661, 2016.

Javaheripi, M., de Rosa, G. H., Mukherjee, S., Shah, S., Religa, T. L., Mendes, C. C. T., Bubeck, S., Koushanfar, F., and Dey, D. LiteTransformerSearch: Training-free neural architecture search for efficient language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. 3d object representations for fine-grained categorization. In *IEEE International Conference on Computer Vision Workshops (ICCV 3dRR-13)*, pp. 554–561, 2013.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

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

Li, C., Peng, J., Yuan, L., Wang, G., Liang, X., Lin, L., and Chang, X. Block-wisely supervised neural architecture search with knowledge distillation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1989–1998, 2020.

Li, L. and Talwalkar, A. Random search and reproducibility for neural architecture search. In *Proceedings of the Thirty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 115, pp. 367–377, 2019.

Lin, M., Wang, P., Sun, Z., Chen, H., Sun, X., Qian, Q., Li, H., and Jin, R. Zen-NAS: A zero-shot nas for high-performance image recognition. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 347–356, 2021.

Liu, J., Cai, J., and Zhuang, B. FocusFormer: Focusing on what we need via architecture sampler. *arXiv preprint arXiv:2208.10861*, 2022.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Mellor, J., Turner, J., Storkey, A. J., and Crowley, E. J. Neural architecture search without training. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 7588–7598, 2021.

Nilsback, M. and Zisserman, A. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP)*, pp. 722–729, 2008.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. V. Cats and dogs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3498–3505, 2012.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, 2019.

Pham, H., Guan, M., Zoph, B., Le, Q., and Dean, J. Efficient neural architecture search via parameters sharing. In *International Conference on Machine Learning (ICML)*, pp. 4095–4104, 2018.

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P. Designing network design spaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10428–10436, 2020.

Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. Regularized evolution for image classifier architecture search. In *The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI)*, pp. 4780–4789, 2019.

Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *CoRR*, abs/1801.04381, 2018.

Shu, Y., Cai, S., Dai, Z., Ooi, B. C., and Low, B. K. H. NASI: label- and data-agnostic neural architecture search at initialization. In *International Conference on Learning Representations (ICLR)*, 2022a.

Shu, Y., Dai, Z., Wu, Z., and Low, B. K. H. Unifying and boosting gradient-based training-free neural architecture search. *arXiv preprint arXiv:2201.09785*, 2022b.

Srinivas, A., Lin, T., Parmar, N., Shlens, J., Abbeel, P., and Vaswani, A. Bottleneck transformers for visual recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16519–16529, 2021.

Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., and Xu, C. ViTAS: Vision transformer architecture search. In *European Conference on Computer Vision (ECCV)*, pp. 139–157, 2022.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.

Tan, M. and Le, Q. V. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, volume 97, pp. 6105–6114, 2019.

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., and Le, Q. V. MnasNet: Platform-aware neural architecture search for mobile. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2820–2828, 2019.

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

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 10347–10357, 2021.

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

Wang, H., Wu, Z., Liu, Z., Cai, H., Zhu, L., Gan, C., and Han, S. HAT: hardware-aware transformers for efficient natural language processing. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 7675–7688, 2020b.

Wang, R., Bai, Q., Ao, J., Zhou, L., Xiong, Z., Wei, Z., Zhang, Y., Ko, T., and Li, H. LightHuBERT: Lightweight and configurable speech representation learning with once-for-all hidden-unit BERT. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pp. 1686–1690, 2022.

Wightman, R. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., and Keutzer, K. FBNet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10734–10742, 2019.

Xu, J., Zhao, L., Lin, J., Gao, R., Sun, X., and Yang, H. KNAS: green neural architecture search. In *International Conference on Machine Learning (ICML)*, volume 139, pp. 11613–11625, 2021a.

Xu, Y., Zhang, Q., Zhang, J., and Tao, D. ViTAE: Vision transformer advanced by exploring intrinsic inductive bias. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 28522–28535, 2021b.

Yang, C., Wang, Y., Zhang, J., Zhang, H., Wei, Z., Lin, Z., and Yuille, A. L. Lite vision transformer with enhanced self-attention. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11988–11998, 2022.

Yin, Y., Chen, C., Shang, L., Jiang, X., Chen, X., and Liu, Q. AutoTinyBERT: Automatic hyper-parameter optimization for efficient pre-trained language models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 5146–5157, 2021.

Yu, J. and Huang, T. S. Universally slimmable networks and improved training techniques. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1803–1811, 2019.

Yu, J., Jin, P., Liu, H., Bender, G., Kindermans, P., Tan, M., Huang, T. S., Song, X., Pang, R., and Le, Q. BigNAS: Scaling up neural architecture search with big single-stage models. In *European Conference on Computer Vision (ECCV)*, volume 12352, pp. 702–717, 2020.

Yuan, L., Tay, F. E. H., Li, G., Wang, T., and Feng, J. Revisiting knowledge distillation via label smoothing regularization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3902–3910, 2020.

Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z., Tay, F. E. H., Feng, J., and Yan, S. Tokens-to-token ViT: Training vision transformers from scratch on ImageNet. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 538–547, 2021.

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. CutMix: Regularization strategy to train strong classifiers with localizable features. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6022–6031, 2019.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random erasing data augmentation. In *AAAI Conference on Artificial Intelligence (AAAI)*, pp. 13001–13008, 2020.

Zhou, Q., Sheng, K., Zheng, X., Li, K., Sun, X., Tian, Y., Chen, J., and Ji, R. Training-free transformer architecture search. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10884–10893, 2022.

A. Statistical Analysis for SNIP with Weight Sharing Strategy in Transformer

We analyze the subnets after the conventional one-shot training since it fairly optimizes the subnets in the whole search space and keeps the consistence of the performance with the model quality. We random sampling 2000 subnets with different model sizes inheriting weights form supernet and find that the SNIP score has 0.86 Kendall correlation with the performance, a littler higher than the 0.85 correlation of the model size and performance. To further leverage the reason of the bad performance of SNIP on the whole search space, we devote ourselves to analyzing the performance of SNIP on the subnets with equal model size.

Table 7. The statistics of the analyzed subnets in Case 1 and Case 2. We group the subnets by their size and make statistics of the number of subnets and the accuracy span from the worst subnet to the best subnet in the group.

Case 1	Group ID	1	2	3	4	5	6	7	8	9	10	11
	Model Num.	12	66	220	495	792	924	792	495	220	66	12
	Acc. Span	0.334	0.542	0.730	0.842	1.008	0.958	0.944	0.820	0.762	0.378	0.336
Case 2	Group ID	1	2	3	4	5	6	7	8	9	10	11
	Model Num.	12	66	220	495	792	924	792	495	220	66	12
	Acc. Span	0.280	0.322	0.332	0.372	0.394	0.382	0.386	0.372	0.296	0.240	0.160

We focus our analysis on two cases:

(Case 1) We fix the embedding dimension as 192, depth as 12, number of heads in the multi-head attention block as 4 in each layer and consider all the subnets with 3.5 or 4 MLP ratio. We group the subnets as the Tab. 7 shows where the group ID is equal to the number of layers with 4 MLP ratio.

(Case 2) We fix the embedding dimension as 192, depth as 12, MLP ratio as 4 in each layer and consider all the subnets with 3 or 4 heads in multi-head attention blocks. We group the subnets as the Tab. 7 shows where the group ID is equal to the number of layers with 4 heads in multi-head attention blocks.

As the Tab. 7 shows, it is worthwhile to find a great distributions of the numbers of heads and especially of the MLP ratios since of the obvious different performance in the subnets with equal model sizes. Besides, that reduces the search space intensively while keeping the same range of model sizes as the whole search space. However, we find the SNIP proxy fails in doing such things as following shows.

The mean Best Rank (mBR) (Chen et al., 2021a) is proposed for measuring the performance of the zero-cost proxy in different sets of networks and emphasizes the actual quality of the best network selected by the proxy which is very suitable for our situations. It can formalized as

$$mBR = \frac{\sum_g (r_g - 1)}{\sum_g (|g| - 1)},$$

where g denotes the groups in our situations, r_g is the actual rank in accuracy of the subnet selected by the proxy as showed in Eq. 3 and $|g|$ is the number of subnets in this group. If the proxy always selects the best network, $mBR = 0$ but $mBR = 1$ for the worst network.

Table 8. The mBR of SNIP in Case 1 and Case 2.

		mBR
Case 1	SNIP	0.986
	SNIP with layer norm.	0.099
Case 2	SNIP	0.825
	SNIP with layer norm.	0.180

As the Tab. 8 shows, the SNIP selects almost the worst subnets in both Case 1 and Case 2. Then we compare the SNIP score in each layer of supernet. We find that there are higher SNIP scores in the lower layers since of the slight higher gradients.

That leads SNIP to select the subnets with wider lower layers under the weight and gradient sharing strategy with supernet. In view of this, we propose the SNIP proxy with layer normalization as the Eq. 13 shows. The small mBRs of the SNIP proxy with layer normalization in Tab. 8 verify its effectiveness.

B. The Greedy Algorithm

The isomer architectures in $\mathcal{A}_g \in G$, where G is the segmentation of the Transformer search space $\mathcal{A}$, have the same embedding dimensions, depth, total number of heads and total MLP ratios in all layers since of the same structures in layers of Transformer. Hence, we only need to consider how to allocate heads and MLP ratios to each layer to approximate the architecture with max $\mathcal{P}^{\text{norm}}_{\text{SNIP}}$ in each $\mathcal{A}_g$. Here, we only show the allocation for heads in Algorithm 1 and the allocation of MLP ratio is similar. It is obvious that when there are only two choices of the number of heads and MLP ratio in each layer, the architecture induced by the greedy algorithm is exact the optimized one.

Algorithm 1 Greedy allocation of heads for isomer architectures

Input: The layer numbers n, total number h of heads, head number lower bound d and head number upper bound u.
Output: Head allocation in each layer alc.
Initialize alc as $alc[l] = d$ for each layer l.
Calculate the number of left heads to allocate as $left = h - n * d$.
for $i = 1$ **to** $left$ **do**
  If $alc[l] < u$, calculate the SNIP score $\mathcal{P}[l]$ of the $(alc[l] + 1)$th head in each layer l with layer normalization via Eq. 13, else $\mathcal{P}[l] = 0$.
  $cur = \arg\max_l \mathcal{P}[l]$
  $alc[cur] = alc[cur] + 1$
end for
Return: alc

C. Regularization & Data Augmentation

The learning capacity of three supernets are significantly different, with parameters ranging from minimum 5.4M to maximum 76M. Therefore, we appropriately decrease regularization and data augmentation for Supernet-Tiny and accordingly increase the magnitudes for Supernet-Base to avoid overfitting or underfitting. The detailed hyper-parameter settings are presented in Tab. 9. The value terminology primarily follows timm (Wightman, 2019).

Table 9. Hyper-parameters of training regularization and data augmentation.

Techniques	Supernet-Tiny	Supernet-Small	Supernet-Base
Weight decay	0.02	0.05	0.05
Label smoothing	0.1	0.1	0.1
Stoch. Depth	✓	✓	✓
Repeated Aug	✓	✓	✓
Mix switch prob	✗	0.5	0.5
Mixup alpha	0	0.8	0.8
Mixup mode	✗	elem	elem
Cutmix alpha	0	1	1
Rand Augment	m9-n2-mstd0.5-inc1	✗	m10-n3-mstd0.5-inc1
AutoAug	✗	v0r-mstd0.5	✗
Erasing prob	0.25	0.25	0.25
Erasing count	1	1	2