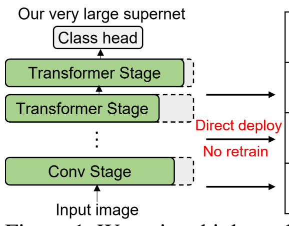
ElasticViT: Conflict-aware Supernet Training for Deploying Fast Vision Transformer on Diverse Mobile Devices

Chen Tang[1*§] Li Lyna Zhang[2*‡] Huiqiang Jiang[2] Jiahang Xu[2] Ting Cao[2]
Quanlu Zhang[2] Yuqing Yang[2] Zhi Wang[1] Mao Yang[2]

[1]Tsinghua University, [2]Microsoft Research

Abstract

Neural Architecture Search (NAS) has shown promising performance in the automatic design of vision transformers (ViT) exceeding 1G FLOPs. However, designing lightweight and low-latency ViT models for diverse mobile devices remains a big challenge. In this work, we propose ElasticViT, a two-stage NAS approach that trains a high-quality ViT supernet over a very large search space that supports a wide range of mobile devices, and then searches an optimal sub-network (subnet) for direct deployment. However, prior supernet training methods that rely on uniform sampling suffer from the gradient conflict issue: the sampled subnets can have vastly different model sizes (e.g., 50M vs. 2G FLOPs), leading to different optimization directions and inferior performance. To address this challenge, we propose two novel sampling techniques: complexity-aware sampling and performance-aware sampling. Complexity-aware sampling limits the FLOPs difference among the subnets sampled across adjacent training steps, while covering different-sized subnets in the search space. Performance-aware sampling further selects subnets that have good accuracy, which can reduce gradient conflicts and improve supernet quality. Our discovered models, ElasticViT models, achieve top-1 accuracy from 67.2% to 80.0% on ImageNet from 60M to 800M FLOPs without extra retraining, outperforming all prior CNNs and ViTs in terms of accuracy and latency. Our tiny and small models are also the first ViT models that surpass state-of-the-art CNNs with significantly lower latency on mobile devices. For instance, ElasticViT-S1 runs 2.62× faster than EfficientNet-B0 with 0.1% higher accuracy.



Device (year)	Model	FLOPs (M)	TOP-1 (%)	Latency (ms)
Pixel 1 (2016)	MobileNetV3	219	75.2	35.0
	ElasticViT-T3	**160**	**75.2**	**29.2**
Pixel 4 (2019)	EfficientNet-B0	390	77.1	55.1
	ElasticViT-S1	**218**	**77.2**	**21.0**
Pixel 6 (2021)	AutoFormer-Tiny	1300	74.7	71.1
	ElasticViT-L3	**806**	**80.0**	**50.5**

Figure 1. We train a high-quality ViT supernet for a wide range of mobile devices. Our discovered ViTs outperform SOTA CNNs and ViTs with higher accuracy, fewer FLOPs and faster speed.

1. Introduction

Vision Transformers (ViTs) have achieved remarkable success in various computer vision tasks [14, 31, 51, 5, 64]. However, the success comes at a significant cost - ViTs are heavy-weight and have high inference latency costs, posing a great challenge to bring ViTs to resource-limited mobile devices [56]. Designing accurate and low-latency ViTs becomes an important but challenging problem.

Neural Architecture Search (NAS) provides a powerful tool for automating efficient DNN design. Recently, two-stage NAS such as BigNAS [59] and AutoFormer [7], decouple training and searching process and achieves remarkable search efficiency and accuracy. The first stage trains a weight-shared supernet assembling all candidate architectures in the search space, and the second stage uses typical search algorithms to find best sub-networks (subnets) under various resource constraints. The searched subnets can directly inherit supernet weights for deployment, achieving comparable accuracy to those retrained from scratch. Such two-stage NAS can eliminate the prohibitively expensive cost for traditional NAS to retrain each subnet, making it a practical approach for efficient deployment.

The success of two-stage NAS heavily relies on the quality of the supernet training in the first stage. However, it's extremely challenging to train a high-quality ViT supernet for mobile devices, due to the **vast mobile diversity**: mobile applications must support a wide range of mobile phones

[*]Equal contribution
[§]Work is done during the internship at Microsoft Research
[‡]Corresponding author, email: lzhani@microsoft.com

with varying computation capabilities, from the latest high-end devices to older ones with much slower CPUs. For instance, Google Pixel 1 runs $4\times$ slower than Pixel 6. As a result, the supernet must cover ViTs that range from tiny size ($<$ 100M FLOPs) for weak devices to large size for strong ones. However, including both tiny and large ViTs results in an overwhelmingly larger search space compared to typical search spaces in two-stage NAS [3, 59, 15]. Training a supernet over such a search space has been known to suffer from performance degradation due to optimization interference caused by subnets with vastly different sizes [11, 60, 63]. While existing works [28, 62, 7, 47] circumvent this issue by manually designing multiple separate normal-sized search spaces, the multi-space approach can be costly. Moreover, there has been limited discussion regarding the root causes of this problem and how to effectively address it.

In this work, we introduce ElasticViT, a novel approach for training a high-quality vision transformer supernet that can efficiently serve both strong and weak mobile devices. Our approach is built upon a single very large search space optimized for mobile devices, containing a wide range of vision transformers with sizes ranging from 37M to 3G FLOPs. This search space is $10^7\times$ larger than typical two-stage NAS search spaces, allowing us to accommodate a broad range of mobile devices with various resource constraints.

We start by investigating the root causes of poor performance when training a ViT supernet over our excessively large search space. We found that the main reason is that prior supernet training methods rely on uniform sampling [7, 15, 59, 3], which can easily sample subnets with vastly different model sizes (e.g., 50M vs. 1G FLOPs) from our search space. This leads to conflicts between subnets' gradients and creates optimization challenges. We make two key observations:*(i) the gradient conflict between two subnets increases with the FLOPs difference between them;* and *(ii) gradient conflict between same-sized subnets can be significantly reduced if they are good subnets.*

Inspired by the above observations, we propose two key techniques to address the gradient conflict issue. First, we propose *complexity-aware sampling* to limit the difference in FLOPs between sampled subnets across adjacent training steps, while ensuring that different-sized subnets within the search space are sampled. We achieve this by constraining the FLOPs level of the sampled subnets to be close to that of the previous step. Furthermore, we employ a multiple-min strategy to sample the nearest smallest subnet based on the FLOPs sampled at each step, thus ensuring performance bounds without introducing a large FLOPs difference with other subnets. Second, we introduce *performance-aware sampling* that further reduces the gradient conflicts among subnets with similar FLOPs. Our method samples subnets with higher potential accuracy at each step from a prior distribution that is dynamically updated based on an exploration and exploitation policy. The policy leverages a memory bank and a ViT architecture preference rule. The preference rule guides the exploration of subnets with wider width and shallower depth, which are empirically preferred by ViT architectures. The memory bank stores historical good subnets for each FLOPs level using prediction loss as a criterion.

Our contributions are summarized as follows:

- We propose ElasticViT to automate the design of accurate and low-latency ViTs for diverse mobile devices. For the first time we are able to train a high-quality ViT supernet over a vast and mobile-regime search space.
- We conduct thorough analysis on the poor-quality supernet trained by existing approaches, and find that uniform sampling results in subnets of vastly different sizes, leading to gradient conflicts.
- Inspired by our analysis, we propose two methods, complexity-aware sampling and performance-aware sampling, to effectively address the gradient issues by sampling good subnets and limiting their FLOPs differences across adjacent training steps.
- Extensive experiments on ImageNet [12] and four mobile devices demonstrate that our discovered models achieve significant improvements over SOTA efficient CNN and ViT models in terms of both inference speed and accuracy. For example, ElasticViT-T3 achieves the same accuracy of 75.2% as MobileNetV3 with only 160 MFLOPs, while be $1.2\times$ faster. This is the first time that ViT outperforms CNN with a faster speed on mobile devices within the 200 MFLOPs range, to the best of our knowledge. ElasticViT-L achieves 80.0% accuracy with 806 MFLOPs, which is 5.3% higher than Autoformer-Tiny [7] while using $1.61\times$ fewer FLOPs. We also prove that ElasticViT substantially enhance the quality of supernet training, resulting in a noteworthy 3.9% accuracy improvements for best-searched models.

2. Related works

Efficient Vision transformers. Many methods have been proposed to design efficient ViTs. They use different ways, such as new architectures or modules [6, 19, 25], better attention operation [34, 36, 31, 24] and hybrid CNN-transformer [57, 17, 33, 52, 16, 9, 26]. Hybrid models usually perform well with small sizes by introducing special operations. For instance, MobileFormer [9] uses a parallel CNN-transformer structure with bidirectional bridge. However, although the FLOPs are reduced, these ViTs still have high latency because of mobile-unfriendly operations such as the bidirectional bridge.

Neural Architecture Search. NAS has achieved an amazing success in automating the design of efficient CNN architectures [40, 18, 4, 59, 3]. Recently, several works apply NAS to find improved ViTs, such as Autoformer [7], S3 [8], ViTAS [46] and ViT-ResNAS [27]. These methods focus on searching models exceeding 1G FLOPs. For small ViTs, HR-NAS [13], UniNet [30] and NASViT [15] search for hybrid CNN-ViTs and achieve promising results under small FLOPs. However, these NAS works mainly optimize for FLOPs without considering the efficiency on diverse mobile devices, which leads to suboptimal performance.

Supernet Training. Early NAS methods [65, 66, 42, 43] are very costly because they need to train and evaluate many architectures from scratch. More recent one-shot NAS methods [18, 4, 29, 10] use weight-sharing to save time. But they still need to retrain the best architecture from scratch for higher accuracy, which is expensive when targeting multiple constraints. To solve this problem, two-stage NAS, such as OFA [3], BigNAS [59] and AutoFormer [7] decouples the training and search. They train a supernet where the good subnets can be directly deployed without retraining. However, they employ uniform sampling to sample subnets, which can lead to subnets with significantly different sizes being sampled in a much larger search space, resulting in gradient conflicts and inferior performance. Our work proposes conflict-aware supernet training to address this issue.

3. Search Space Design and Training Analysis

3.1. Search Space Design

Mobile-friendly ViTs. While many works aim to design ViT models with high accuracy and small FLOPs, we observe that models with small FLOPs may have high latency on mobile devices. For example, NASViT-A0 [15] has fewer FLOPs than MobileNetV3, but it runs $2\times$ slower on a Pixel4 phone. MobileFormer [9] has only 52M FLOPs but runs $5.5\times$ slower. This is because these models incorporate effective but *mobile-unfriendly operations* that reduce FLOPs.

Our goal is to design accurate ViTs that can achieve low-latency on mobile devices. To achieve this, we draw inspiration from recent works [16, 15] and construct our search space based on mobile-friendly CNN-ViT architecture as shown in Table 1. In CNN stage, we use MobileNetv2 [44] (MBv2) and MobileNetv3 [20] (MBv3) blocks. In ViT stage, we make key modifications based on NASViT attentions for better efficiency. We remove the slow talking heads [45] and use Hswish instead of Gelu. We opt not to employ shifted window attention in Swin [31], because it does not help much when the input size is small. We measure the latency on real devices. Our attention can speed up latency by $>2\times$, making the transformer block more efficient.

A very large search space. Unlike previous ViT NAS works [7, 46] that focused primarily on large models exceeding 1G FLOPs, our search space must accommodate a wide range of ViT configurations, from tiny to large, to meet the demands of diverse mobile devices. For instance, a high-end device like the Pixel 6 can handle a large ViT model with 500M FLOPs to meet latency constraint of $\sim$30ms, whereas a less powerful device such as the Pixel 1 requires a tiny

Table 1. Our very large search space to support both weak and strong mobile devices. It's $10^7\times$ larger than that in two-stage NAS. Tuples of three values represent the lowest value, highest, and steps.

Stage	Depths	Channels	Kernel size (V scale)	Expansion ratio	Stride
Conv 3×3	1	(16, 24, 8)	3	-	2
MBv2 block	1-2	(16, 24, 8)	3, 5	1	1
MBv2 block	2-5	(16, 32, 8)	3, 5	3, 4, 5, 6	2
MBv3 block	2-6	(16, 48, 8)	3, 5	3, 4, 5, 6	2
Transformer	1-5	(48, 96, 16)	2, 3, 4	2, 3, 4, 5	2
Transformer	1-6	(80, 160, 16)	2, 3, 4	2, 3, 4, 5	1
Transformer	1-6	(144, 288, 16)	2, 3, 4	2, 3, 4, 5	2
Transformer	1-6	(160, 320, 16)	2, 3, 4	2, 3, 4, 5	2
MBPool	-	1984	-	6	-
Input resolution	128, 160, 176, 192, 224, 256				

model with $<$100M FLOPs to meet the same constraint.

Table 1 presents the final search space. We add many small choices for each block dimensions to include tiny ViTs. We also make several key designs to cover potentially good subnets based on previous works. Specifically, we follow [37] and increase the maximun width choices and decrease the depth choices for ViT stages. We follow LeViT [16] and allow V matrix to have larger expansion ratio of $\{2, 3, 4\}$ (i.e., V scale). We also allow larger expansion ratios of $\{2, 3, 4, 5\}$ for MLP layers, because they are not redundant when using a typical ratio of 2.

In total, our search space covers a wide range of ViT subnets with varying sizes, from 37 to 3191 MFLOPs. It contains an enormous 1.09×10^{17} subnets, which is a significant increase of **$10^7\times$ larger** than a typical search space in two-stage NAS (refer to Appendix. C for details). This presents new challenges in training a high-quality supernet over such a large search space.

3.2. Analysis of Training a Very Large Supernet

Supernet training differs from standard single network training in that all the subnets share weights for their common parts. The shared weights may receive conflicting gradients that lead to different optimization directions, which lowers the final accuracy. Several techniques have been proposed to mitigate this issue [59, 54, 15], have demonstrated success in training high-quality supernet over typical search spaces. Among them, sandwich rule is essential to ensure performance lower and upper bounds. Specifically, it samples a min, max and two random subnets per iteration.

However, when training a ViT supernet over our vast search space in Table 1, we observe significant accuracy drop using previous best practices [54, 15]. Specifically, we use sandwich rule and follow the same training receipts in NASViT [15]. Fig. 2(a) compares the accuracy of 20 random subnets achieved by inheriting supernet weights to retraining on ImageNet. Compared to retraining, models derived from the supernet experience an accuracy drop with up to 8%.

Analysis. Compared to a typical ViT supernet, our supernet includes many tiny subnets and the sizes between subnets can differ greatly. We randomly sample subnets with MFLOPs ranging from 50 to 1200 from our supernet, and compute

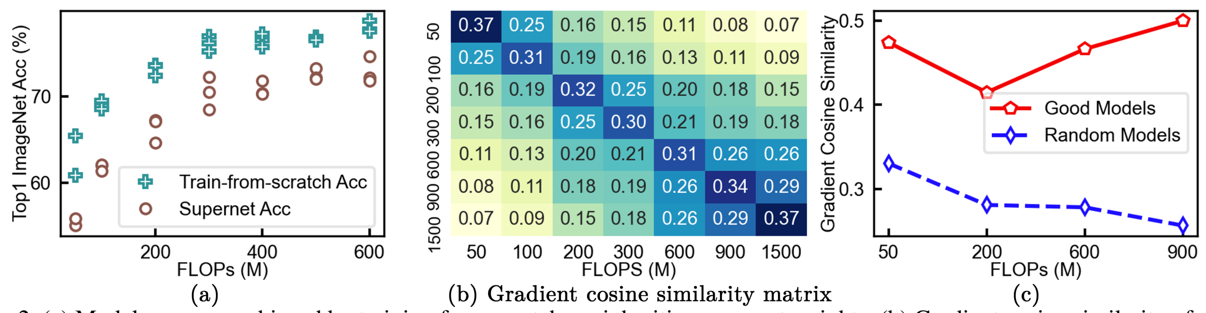


Figure 2. (a) Model accuracy achieved by training from scratch vs. inheriting supernet weights; (b) Gradient cosine similarity of models with different FLOPs; (c) Under the same FLOPs level, good models share more similar gradients than random sampled models.

the cosine similarity of shared weights' gradient between each pair under the same batch of training data. A lower similarity indicates larger differences in gradients and thus more difficulty in training the supernet well. Fig. 2(b) shows that *gradient similarity of shared weights between two subnets is negatively correlated with their FLOPs difference* (**Observation#1**). Subnets with similar FLOPs achieve the highest gradient similarity, while the similarity is close to 0 if there is a large FLOPs difference between two subnets.

Besides FLOPs difference, subnet quality may also affect gradient similarity of shared weights. If a poor subnet is sampled and trained, it would disturb the weights of good subnets. To verify this hypothesis, we randomly sample 50 subnets with same level FLOPs and compute the shared weights gradient cosine similarity between top subnets and randomly-sampled subnets. Fig. 2(c) suggests that *gradient similarity between same-size subnets can be greatly improved if they are good subnets* (**Observation#2**).

4. Methodology

Inspired by the observations in Sec. 4.3, we propose two methods to address the gradient conflict issue. (i) We introduce complexity-aware sampling to restrict the FLOPs difference between adjacent training steps, as large difference incurs gradient conflict problem (Sec. 4.2). (ii) Since "good subnets" can enhance each other, which further reduce gradient conflicts, we propose performance-aware sampling to ensure the training process can sample good subnets as much as possible (Sec. 4.3).

4.1. Preliminary

For a search space $\mathcal{A}$, two-stage NAS aims to train a weight-sharing network (supernet) over $\mathcal{A}$ and jointly optimize the parameters of all subnets. Since it is infeasible to train all subnets in $\mathcal{A}$, this is often achieved by the widely-used *sandwich rule*, which samples two types of subnets at each training step: **(i)** an expectation term approximated by a sampled subnet set through a prior distribution Γ over the search space $\{s_m | s_m \sim \Gamma(\mathcal{A})\}_{m=1}^{M}$ and **(ii)** a fixed subnet set $s_n \in \mathcal{S}$. Formally, the supernet training can be framed as the following optimization problem:

$$\arg\min_{w} \left[\mathbb{E}_{s_m \sim \Gamma(\mathcal{A})} \mathcal{L}_D \left(f\left(w_{s_m} \right) \right) + \sum_{s_n \in \mathcal{S}} \mathcal{L}_D (f (w_{s_n})) \right] \quad (1)$$

where w is the shared weights for all subnets, $f(\cdot)$ denotes the neural network, $\mathcal{L}_{\mathcal{T}}$ is the loss on the training set D and w_s is the exclusive weights of subnet s.

Without loss of generality, there are two types of subnets in any space – *the random subnets that each dimensions are sampled between maximum and minimum settings*; and *the smallest and largest subnets that each dimensions are the minimum and maximum settings, respectively*. In recent works [59, 55, 15], sandwich rule often approximates the first term in Eq. 1 by randomly sampling M=2 subnets from a uniform distribution Γ. In the second term, $\mathcal{S}$ typically consists of the largest subnet s_l and the smallest subnet s_s.

We now revisit the effectiveness of applying sandwich rule on training our mobile-specialized supernet in Table 1. Obviously, it can easily cause gradient conflicts due to two reasons. First, it always sample the smallest (37M FLOPs), biggest (3191 MFLOPs) and 2 random subnets, which results in a significant FLOPs difference and often causes the gradient similarity to be close to 0 (Fig. 2(b)). Second, the size and quality of the 2 randomly sampled subnets cannot be guaranteed, which exacerbates the issue.

4.2. Complexity-aware Sampling

In this section, we introduce complexity-aware sampling to mitigate the gradient conflicts by large FLOPs difference, while ensuring that different-sized subnets can be trained.

Adjacent Step Sampling For Uniform Subnets. Due to the extremely large space, randomly sampling M subnets can result in significant FLOPs differences (*see Appendix.C for detailed illustrations*) both within the same training step and across different training steps. Therefore, we propose to constrain the FLOPs of M random subnets for each training step. As shown in Fig. 3, we simply constrain the subnets within the same training step to have the same level of FLOPs.

We use FLOPs to represent the complexity metric, which can trivially be generalized to other metrics (e.g., latency).

However, it is non-trivial to effectively constrain the FLOPs differences between different training steps, as it requires ensuring that subnets of varying sizes can be trained to support diverse resource constraints. We propose adjacent step sampling to gradually change the FLOPs level.

Specifically, we define a set of gradually increased complexity levels $C_1, ..., C_K$ (e.g., 100, 200, ..., 800 MFLOPs), which cover a range of ViT models from tiny to large. Suppose step t-1 samples the i^{th} complexity level $C_i^{(t-1)}$, and step t samples M subnets $s^{(t)}$ under the j^{th} complexity level of $C_j^{(t)}$. To satisfy the adjacent step sampling, the FLOPs distance between these two steps must satisfy the following:

$$g\left(s^{(t)}; C_i^{(t-1)}\right) = |C_j^{(t)} - C_i^{(t-1)}| = 0 \tag{2}$$

To satisfy the equation, we offer $C_j^{(t)}$ three options: $C_{i-1}^{(t-1)}, C_i^{(t-1)}, C_{i+1}^{(t-1)}$, representing the choices of *decreasing the FLOPs level by one*, *maintaining the current FLOPs level*, or *increasing the FLOPs level by one*, respectively.

Remove the Biggest Subnet. Since the biggest subnet has 3191 MFLOPs in our search space, it naturally introduces a large FLOPs difference with our considered complexity levels, which can cause gradient conflict at each step. Empirically, we find that removing the largest subnet stabilizes the training process and improves overall performance.

Use Multiple Hierarchical Smallest Subnets (HSS). Since the smallest subnet has only 37 MFLOPs, sampling at the large complexity range ($C_i \geq 500$ MFLOPs) can also introduce large FLOPs difference at each step. Unlike the biggest subnet, the smallest subnet decides the performance lower bound of the whole space [50], which cannot be removed simply. Therefore, instead of sampling the min subnet with only 37 MFLOPs at each step, we sample a nearest min subnet from the *hierarchical smallest subnets (HSS) set* $\hat{\mathbf{S}} = \{s_n\}_{n=1}^N$. HSS set includes N=3 pre-defined subnets with discrepant complexity. At step t, when sampling around a complexity level $C^{(t)}$, we select a subnet $s_n \in \hat{\mathbf{S}}$ whose complexity is closest to it as the smallest subnet, as in Eq. 5.

In our experiments, we conduct empirical analysis and select the N=3 smallest subnets as the HSS set $\hat{\mathbf{S}}$. As shown in Fig. 3, these subnets include the original 37 MFLOPs subnet (min$_1$), as well as a 160 MFLOPs subnet (min$_2$) and a 280 MFLOPs subnet (min$_3$), which are sampled and added as the second and third smallest subnets, respectively.

Discussion for HSS set. The multiple "smallest" subnets in HSS set logically partition the whole search space into N hierarchical sub-spaces (Fig. 3). This is differs from previous methods[7, 8], which manually divide the space into separate sub-spaces and train them individually. The HSS set offers two advantages: *(i)*: it enables unified weight-sharing across the entire space, allowing small subnets to benefit from large subnets. Moreover, it has been proven that incorporating small models into larger ones can significantly enhance small subnets' performance[2]. *(ii)*: It does not rely on any heuristic space re-design or strong relationship assumption between dimensions (*e.g.,* linear correlation in [8]), which enhances the universality of our method.

Optimization Objective. With applying complexity-aware sampling, we reformulate Eq. 1 as the follows:

$$\arg\min_{w} \left[\sum_{s_m^{(t)} \in \mathbf{U}}^{M} \mathcal{L}_D\left(f(w_{s_m^{(t)}})\right) + \sum_{s_n \in \hat{\mathbf{S}}} \sigma(s_n, C_j^{(t)}) \mathcal{L}_D\left(f(w_{s_n})\right) \right], \tag{3}$$

where t denotes the current training step, $\mathbf{U}$ is the stochastic subnet set containing M=3 uniform subnets, in which each subnet has the FLOPs level of $C_j^{(t)}$ for step t:

$$\mathbf{U} = \left\{ s_m^{(t)} | s_m^{(t)} \sim \Gamma(\mathcal{A}) \; \& \; g'\left(s_m^{(t)}; C_j^{(t)}\right) == 0 \right\}_{m=1}^{M}, \tag{4}$$

and $\sigma(\cdot)$ selects the nearest smallest subnet from HSS:

$$\sigma(s_n, C_j^{(t)}) = \begin{cases} 1 & \text{if } s_n \text{ is the nearest min that is larger than } C_j^{(t)} \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

Note that we still use the notion Γ since the prior distribution will be determined in Sec. 4.3.

4.3. Performance-aware Sampling

In Sec. 3.2, we observe that top-performing subnets can further alleviate the gradient conflict issue. Inspired by this, we introduce a performance-aware sampling to sample subnets with potentially higher accuracy.

Specifically, for the $M = 3$ same-sized subnets in $\mathbf{U}$ (as defined in Eq. 4), we aim to sample from a new distribution that favors good subnets, rather than a uniform distribution with random performance. To achieve this, we propose an exploration and exploitation policy that constructs the new distribution based on quality-aware memory bank and path preference rule. The quality-aware memory bank is used to exploit historical good subnets with a probability of q, while preference rule explores subnets with wider width and shallower depth with a probability of $1 - q$.

Quality-aware Memory Bank. As shown in Fig. 3, *memory bank* stores the historical up-to-date good subnets for each FLOPs level. Specifically, the good subnets are identified through comparing cross-entropy loss on the current min-batch. Suppose step t sample the j^{th} FLOPs level of C_j, B_j denotes the good subnets for FLOPs of C_j at step t, the prior distribution Γ can be as a mixture distribution of the dynamic memory bank and other unexplored subnets. From this perspective, the expectation of subnet $s_m^{(t)}$ is

$$\mathbb{E}_{s_m^{(t)} \sim \Gamma(\mathcal{A})} = q \cdot U(\mathrm{B}_j) + (1 - q) \cdot U(\hat{\mathcal{A}}_{C_j}), \tag{6}$$
$$\text{where } \; \mathrm{B}_j \cup \hat{\mathcal{A}}_{C_j} = \mathcal{A}_{C_j}$$

where $\mathcal{A}_{C_j}$ denotes all the subnets with FLOPs level of C_j in search space $\mathcal{A}$. $U(\hat{\mathcal{A}}_{C_j})$ is the uniform distribution of unexplored subnets, which will be further regulated by the path preference rule in next section.

At the early training, a relatively small value (i.e., 0.2) of q is applied so that uniform sampling dominates the training to exploring promising subnets. As training proceeds, the memory banks are gradually filled, at which q is also gradually increased to exploit these memorized good subnets.

Memory bank replacement strategy. We adopt the *Worst-Performing* strategy to replace the subnet in the memory bank. When the current subnet outperforms the worst-performing subnet in the memory bank, the worst-performing subnet is replaced by the current subnet.

Path Preference Rule. In Eq. 6, when exploring unvisited subnets from $\hat{\mathcal{A}_{C_j}}$, we propose to sample ViT-preferred architectures that are more likely to achieve higher accuracy. Our approach is inspired by recent studies [37, 41], which found that in pure ViT models, the later Transformer stages tend to prefer wider channels over more layers compared to CNN models. We empirically verify that this conclusion holds true in the hybrid CNN-Transformer NAS space (see Appendix), which motivates us to incorporate this preference into the super-network training to filter out inferior subnets.

Specifically, when a subnet $s^{(t)} = \{o^{(t)}_{width}, o^{(t)}_{depth}\}$ is sampled (*i.e.,* through uniform distribution of second term in Eq. 6), we expect it to have *wider widths and shallower depths in Transformer stages*, where $o^{(t)}_{width}$ and $o^{(t)}_{depth}$ are the width and depth dimension (*i.e.,* the dimension of interests), respectively.

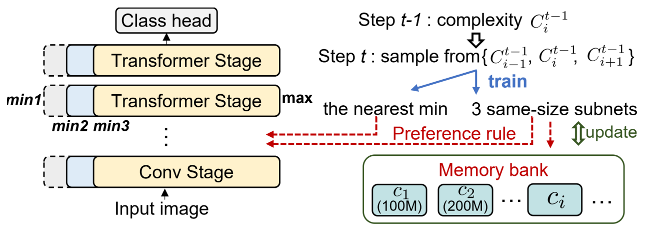
However, it's non-trivial to determine whether a ViT is a deep and narrow model or a shallow and wide model. Our approach involves quantifying a subnet's FLOPs distribution in terms of the depth and width dimensions by comparing it with an anchor model. This enables us to determine the subnet's preference in terms of depths and widths. Specifically, suppose the anchor model is $s^{ac} = \{o^{ac}_{width}, o^{ac}_{depth}\}$ and the sampled subnet is $s^{(t)} = \{o^{(t)}_{width}, o^{(t)}_{depth}\}$, our first step is to generate a new subnet $\overline{s^{(t)}} = \{\mathbf{o^{ac}_{width}}, o^{(t)}_{depth}\}$ by aligning all dimensions to anchor model except the depth dimension, and another new subnet $\widehat{s^{(t)}} = \{o^{(t)}_{width}, \mathbf{o^{ac}_{depth}}\}$ by aligning all dimensions to anchor model except widths. Then, we compute the FLOPs differences between two new subnets and the anchor model:

$$\Phi_a(o^{(t)}_{width}) = \text{FLOPs}(\overline{s^{(t)}}) - \text{FLOPs}(s^{ac}), \tag{7}$$

$$\Phi_b(o^{(t)}_{depth}) = \text{FLOPs}(\widehat{s^{(t)}}) - \text{FLOPs}(s^{ac}). \tag{8}$$

If $\Phi_a \geq \Phi_b$, subnet $s^{(t)}$ is considered to have wider widths and shallower depths for transformer stages, which adheres to path preference rule, and we will train it. Otherwise, we resample a new subnet and repeat the above steps.

For simplicity, we omit the remaining dimensions (e.g., kernel size, expansion ratio, etc.) since they are not the dimensions of interest.



(a) Our very large supernet (b) Conflict-aware supernet training

Figure 3. The overview of our proposed confict-aware supernet training. At step t, we first sample a target FLOPs that is close to that at step t-1, then we sample 3 subnets with same level of FLOPs and find a nearest min subnet to update supernet weights.

The quality of the anchor model s^{ac} can impact the validity of these comparisons. We conjecture that the memory bank captures such preferences and thus select the subnet with minimal loss in the memory bank as the anchor model.

4.4. Overall Supernet Training Process

Fig. 3 shows the supernet training process that uses both complexity-aware and performance-aware sampling techniques. At step t, we choose a FLOPs level $C^{(t)}_j$ from $\{C^{(t-1)}_{i-1}, C^{(t-1)}_i, C^{(t-1)}_{i+1}\}$ that is close to $C^{(t-1)}_i$ at step $t-1$. Then, we sample 1 smallest subnet and M=3 stochastic subnets for training. The smallest subnet is the nearest one from the HSS set with FLOPs closest to $C^{(t)}_j$. The M=3 subnets have FLOPs equal to $C^{(t)}_j$. They are either from the memory bank B_j with probability q or based on the path preference rule with probability 1-q. We compute and accumulate the gradients for above 4 subnets. Then, we use the accumulated gradients to update supernet parameters.

5. Evaluation

Setup. We apply our proposed techniques to train our very large ViT supernet for 600 epochs. The complexity levels used in the training are set to $\{100, 200, 300, 400, 500, 700, 900, 1200\}$ MFLOPs, which are suitable for mobile-regime ViTs. The other training setting and hyper-parameters follows the existing best practices [15, 54]. We list the detailed numbers in supplementary materials.

Then, we evaluate the effectiveness of our trained ViT supernet on four mobile phones with varying resource levels: weak (Pixel1), neutral (Pixel4, Xiaomi11), and strong (Pixel6). For each device, we set a range of latency constraints and use nn-Meter [61] to build a latency predictor for efficient search. We use the evolutionary search method in OFA [3] to search 5k subnets for each latency constraint.

5.1. Main Results on ImageNet

Comparison with efficient CNN and ViT models. Table 2 reports the comparison with state-of-the-art models including both strong CNNs and recent efficient ViTs. Remark-

Table 2. ElasticViT performance on ImageNet-1K [12] with comparison to state-of-the-art efficient CNN and ViT models. We group models based on the hardware they are suited according to FLOPs. *: latency is measured on each group's corresponding hardware.

Model	MFLOPs	Top-1 Acc	Latency*	Type
Tiny models:<100 MFLOPs for weak Pixel1 phone				
ShuffleNet-V2 x0.5 [32]	41	60.3	8.7 ms	CNN
ElasticViT-T0	**37**	**61.1**	**8.5 ms**	ViT NAS
MnasNet-x0.35 [48]	63	64.1	13.5 ms	CNN NAS
MobileFormer [9]	52	68.7	176.6 ms	ViT
ElasticViT-T1	62	67.2	**13.1 ms**	ViT NAS
Tiny models: 100∼150 MFLOPs for weak Pixel1 phone				
ShuffleNet-V2 1× [32]	146	69.4	24.3 ms	CNN
Cream [39]	114	72.8	24.3 ms	CNN NAS
FBNet-v2 [53]	126	73.2	22.5 ms	CNN NAS
MobileNet-V3 x0.75 [20]	155	73.3	26.4 ms	CNN NAS
MobileFormer [9]	96	72.8	216.3 ms	ViT
ElasticViT-T2	**119**	**73.8**	**22.4 ms**	ViT NAS
ElasticViT-T3	160	**75.2**	**29.2 ms**	ViT NAS
Small models: 200∼350 MFLOPs for neutral Pixel4 phone				
MobileNet-v3 x1.0 [20]	219	75.2	24.2 ms	CNN NAS
FBNet-v2 [53]	238	76.0	22.1 ms	CNN NAS
OFA #25 [3]	230	76.4	25.2 ms	CNN NAS
BigNAS-S [59]	242	76.5	32.6 ms	CNN NAS
MobileFormer [9]	214	76.7	415.1 ms	ViT
HR-NAS-A [13]	267	76.6	-	ViT NAS
ElasticViT-S1	**218**	**77.2**	**21.0 ms**	ViT NAS
GreedyNAS-v2 [21]	324	77.5	65.4 ms	CNN NAS
LeViT-128S [16]	305	76.6	30.5 ms	ViT
HR-NAS-B [13]	325	77.3	-	ViT NAS
ElasticViT-S2	**318**	**78.6**	**29.6 ms**	**ViT NAS**
Medium models: 350∼500 MFLOPs for neutral Pixel4 phone				
EfficientNet-B0 [49]	390	77.1	55.1 ms	CNN NAS
BigNAS-M [59]	418	78.9	64.3 ms	CNN NAS
MobileViT-XXS [33]	364	69.0	84.1 ms	ViT
LeViT-128 [16]	406	78.6	40.2 ms	ViT
MobileViTv3-0.5 [52]	481	72.3	96.7 ms	ViT
ElasticViT-M	**415**	**79.1**	**37.4 ms**	ViT NAS
Large models: ≥ 500 MFLOPs for strong Pixel6 phone				
MAGIC-AT [58]	598	76.8	37.9 ms	CNN NAS
BigNAS-L [59]	586	79.5	45.7 ms	CNN NAS
MobileViTv2-0.5 [34]	500	70.2	56.5 ms	ViT
UniNet-B0 [30]	560	79.1	53.9 ms	ViT NAS
ElasticViT-L1	**516**	**79.4**	**33.1 ms**	ViT NAS
EfficientNet-B1 [49]	700	79.1	49.9 ms	CNN NAS
EdgeViT-XXS [36]	600	60.0	69.6 ms	ViT
MobileViT-XS [33]	986	74.8	84.4 ms	ViT
Autoformer-Tiny [7]	1300	74.7	71.1 ms	ViT NAS
ViTAS-Twins-T [47]	1400	79.4	-	ViT NAS
ElasticViT-L2	**704**	**79.8**	**43.8 ms**	ViT NAS
ElasticViT-L3	**806**	**80.0**	**50.5 ms**	ViT NAS

ably, for the first time, we utilize two-stage NAS to deploy lightweight and low-latency ViT models ranging from 37 to 800 MFLOPs, enabling us to bring accurate ViTs to a wide range of mobile devices. Without retraining or finetuning, our discovered subnets ElasticViT models significantly outperform all evaluated ViT and CNN baselines.

First, *our models significantly outperform prior ViTs that are designed for mobile devices.* For tiny ViTs, our ElasticViT-T3 achieves 75.2% accuracy under only 160 MFLOPs, which is 2.9% better than MobileNetV3x0.75 in terms of similar FLOPs. For medium-sized ViTs, ElasticViT-M achieves 79.1% accuracy under 415 MFLOPs, significantly outperforming existing mobile ViTs with 0.5% and 4.8% higher accuracy than LeViT and MobileViTv3 [52], respectively. For large ViT models where ImageNet classification accuracy saturates, ElasticViT-L3 still has 0.6% and 5.3% accuracy improvement compared with Autoformer [7] and ViTAS [47] with 1.7× and 1.6× fewer FLOPs. Furthermore, our ViT models not only achieve high accuracy but also demonstrate fast real inference latency, making them practical for deployment on resource-constrained mobile phones. This sets them apart from other mobile ViT approaches that solely focus on reducing FLOPs. For instance, with only 214 MFLOP, MobileFormer [9] has a slow latency of 415.1 ms on Pixel4, which is 17.2× slower than MobileNetV3 and 19.7× slower than our ElasticViT-S1.

Second, *our models also surpass lightweight CNNs with higher accuracy and lower latency.* For instance, ElasticViT-T0 achieves 61.1% accuracy with only 37 MFLOPs, which outperforms ShuffleNetv2x0.5 with 0.8% higher accuracy. ElasticViT-T3 achieves the same accuracy of 75.2% as MobileNetV3 with only 160 MFLOPs, while be 1.2× faster. This is the first time that ViT outperforms CNN with a faster speed on mobile devices within the 200 MFLOPs range.

Deploying efficient ViTs for diverse mobile phones. Now we apply our ViT supernet to get different specialized subnets for diverse mobile devices. As a baseline for comparison, we choose OFA [3], which represents the state-of-the-art hardware-aware NAS for delivering efficient mobile-regime CNNs. Instead of relying solely on FLOPs to measure on-device efficiency, we use real inference latency for comparison. To ensure a fair comparison, we adopt the same approach as OFA and build a latency predictor for each of our test devices. We conduct an evolutionary search to find the optimal CNN subnets for each device. Note that we didn't compare with ViT NAS baselines, as existing works focus on large ViTs, whose supernets are out of range for searching low-latency models for mobile devices.

Fig. 4 presents the latency-accuracy curve of our ViTs compared to OFA on diverse mobile devices. Our discovered ViT models consistently outperform OFA under various latency constraints, on both weak and strong mobile devices. Notably, our models are the first ViT models to achieve real-time inference latency on mobile devices without compromising accuracy. These results demonstrate the effectiveness of our approach and highlight the potential of transformers for efficient, high-performance models on mobile devices.

5.2. Ablation Study

We now conduct ablation studies to evaluate 1) how each of our techniques can improve supernet training; 2) how our techniques mitigate the gradient conflict issues; and 3) the performance of our searched model compared to retraining.

Ablation study on each technique. Table 3 shows the accuracy of the best-searched models under different supernet training techniques. We start with the baseline supernet that is trained using the original sandwich rule. Then we apply our techniques one by one to enhance the supernet training. We keep all the other training settings and search process

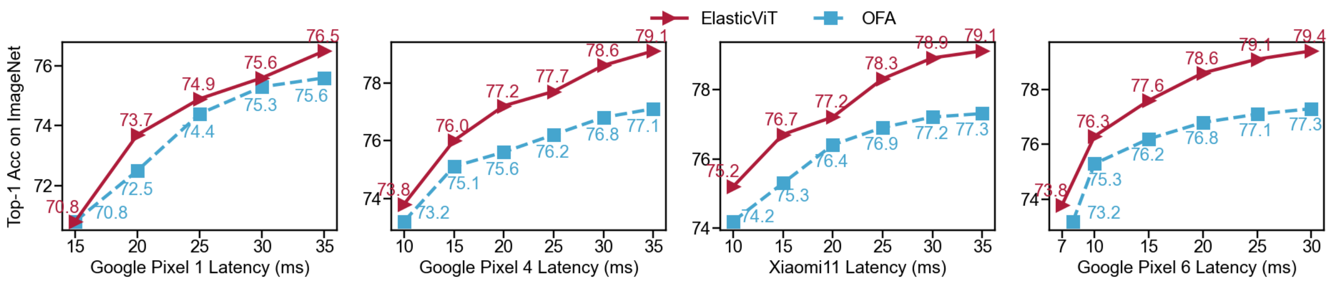


Figure 4. Under the same latency constraint, the discovered ViTs by ElasticViT surpass state-of-the-art mobile CNNs on diverse mobile devices. From left to right: old and weak devices to latest and strong devices.

Table 3. Ablation study results on ImageNet. We show the top1 accuracy of best searched models for each case. Note that *Multiple min* is applied on top of *Adjacent sampling*; *Perf-aware sampling* is applied on top of both *Adjacent sampling* and *Multiple min*.

Method	MFLOPs							
	100	200	300	400	500	600	700	800
Baseline (Sandwich rule)	69.9	74.2	76.5	77.4	77.8	78.3	78.7	79.0
Adjacent step sampling	73.2	75.6	76.7	77.5	78.2	78.3	78.7	79.0
+Multiple min (HSS)	72.8	76.7	78.4	79.0	79.3	79.4	79.6	79.7
++Perf-aware sampling	**73.8**	**77.2**	**78.6**	**79.1**	**79.4**	**79.6**	**79.8**	**80.0**

consistent for a fair comparison.

Table 3 illustrates the effectiveness of our proposed techniques in enhancing supernet training over a vast ViT search space. Both adjacent step sampling and multiple min strategy effectively controls the FLOPs differences among trained subnets, resulting in substantial top-1 accuracy gains of up to 3.3%. Furthermore, by further using performance-aware sampling to train good subnets, we are able to further improve the best-searched ViTs by up to 1% accuracy.

The effectiveness of mitigating gradient conflicts. Our accuracy improvements primarily stem from the effective mitigation of gradient conflicts. To validate this, we conduct experiments on two supernets trained with different approaches: one using our proposed techniques and one using the sandwich rule as a baseline. We freeze the weights for both supernets and study the gradients of different subnets under the same batch of training data. We randomly sample 50 subnets under three levels of FLOPs: 50M, 200M, and 600M. We compute the cosine similarity of shared weights' gradient between each pair of subnets under the same FLOPs. A higher cosine similarity indicates less gradient conflict.

Table 4 shows the average gradient similarity between subnets on both supernets. Compared to vanilla sandwich rule, we can significantly improve the gradient similarity for both random and good subnets, suggesting that our method can efficiently mitigate the gradient conflicts.

Comparison with training from scratch. High-quality supernet training can ensure that subnets achieve comparable accuracy as those trained from scratch. We retrain each subnet with a batch size of 512 on 8 Nvidia V100 GPUs, following the same training settings as LeViT [16]. Table 5 compares the accuracy on ImageNet. These selected subnets can achieve even higher accuracy by directly inheriting weights from our supernet, with up to 2% improvement. Interestingly, we notice that larger ViT models achieve comparable accuracy to retraining, while tiny and small ViTs (<500 MFLOPs) can benefit more from supernet training.

Table 4. Gradient cosine similarity between subnets in supernet trained with different methods. The "good" subnets refer to the top10 subnets chosen from the randomly sampled 50 subnets.

Method	MFLOPs					
	50		200		600	
	Random	Good	Random	Good	Random	Good
Sandwich rule	0.37	0.47	0.32	0.41	0.31	0.46
Ours	**0.50**	**0.56**	**0.51**	**0.67**	**0.51**	**0.67**

Table 5. Best-searched ViT top-1 accuracy of inheriting supernet's weights vs. retraining from scratch.

Method	MFLOPs							
	100	200	300	400	500	600	700	800
Train from scratch	73.4	75.3	76.6	77.7	79.0	79.5	**79.9**	80.0
ElasticViT	**73.8**	**77.2**	**78.6**	**79.1**	**79.4**	**79.6**	79.8	80.0

Table 6. Transfer learning results on downstream image classification datasets. We measure the latency on Pixel 4.

Model	Latency	CIFAR10	CIFAR100	Food-101	Flowers	Pets
MobileNetV3	24.2 ms	97.1	83.3	86.8	94.3	87.7
ElasticViT-S1	**21.0 ms**	**97.5**	**86.1**	**87.2**	**94.3**	**92.1**
LeViT-128S	30.5 ms	96.8	85.0	73.6	86.2	90.1
ElasticViT-S2	**29.6 ms**	**97.5**	**86.9**	**88.3**	**95.2**	**92.9**
EfficientNet-B0	55.1 ms	97.9	86.9	89.1	92.4	92.2
LeViT-128	40.2 ms	97.8	86.6	80.8	86.2	92.2
ElasticViT-M	**37.5 ms**	**97.9**	**87.0**	88.8	**95.6**	**93.3**

5.3. Transfer Learning

We transfer ElasticViT to a list of commonly used transfer learning datasets: 1) CIFAR10 and CIFAR100 [23]; 2) fine-grained classification: Food-101 [1], Oxford Flowers [35] and Pets [38]. We take the pretrained checkpoints on ImageNet and fine-tune on new datasets. We closely follow the hyper-parameter settings in GPipe [22]. The results are summarized in Table 6. Compared to existing efficient CNNs and ViTs, our ElasticViT models achieves significantly better accuracy with fast inference speed on Pixel 4.

6. Conclusion

In this paper, we propose ElasticViT, a two-stage NAS approach that trains a high-quality supernet for deploying accurate and low-latency vision transformers on diverse mobile devices. Our approach introduces two key techniques to address the gradient conflicts issue by constraining FLOPs differences among sampled subnets and sampling potentially good subnets, greatly improving supernet training quality. Our discovered ViT models outperfom prior-art efficient CNNs and ViTs on the ImageNet dataset, establishing new SOTA accuracy under various of latency constraints.

References

[1] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 8

[2] Han Cai, Chuang Gan, Ji Lin, et al. Network augmentation for tiny deep learning. In *Proc. of ICLR*, 2022. 5

[3] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 2, 3, 6, 7

[4] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 2, 3

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ICLR*, 2021. 1

[6] Chun-Fu Richard Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *Proc. of ICCV*, 2021. 2

[7] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. In *Proc. of ICCV*, 2021. 1, 2, 3, 5, 7

[8] Minghao Chen, Kan Wu, Bolin Ni, Houwen Peng, Bei Liu, Jianlong Fu, Hongyang Chao, and Haibin Ling. Searching the search space of vision transformers. In *NeurIPS*, 2021. 2, 5

[9] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. Mobile-former: Bridging mobilenet and transformer. In *Proc. of CVPR*, 2022. 2, 3, 7

[10] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *Proc. of ICCV*, 2021. 3

[11] Yuanzheng Ci, Chen Lin, Ming Sun, Boyu Chen, Hongwen Zhang, and Wanli Ouyang. Evolving search space for neural architecture search. In *Proc. of ICCV*, 2021. 2

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009. 2, 7

[13] Mingyu Ding, Xiaochen Lian, Linjie Yang, Peng Wang, Xiaojie Jin, Zhiwu Lu, and Ping Luo. Hr-nas: Searching efficient high-resolution neural architectures with lightweight transformers. In *Proc. of CVPR*, 2021. 3, 7

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1

[15] 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 *Proc. of ICLR*, 2022. 2, 3, 4, 6

[16] Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Herve Jegou, and Matthijs Douze. Levit: A vision transformer in convnet's clothing for faster inference. In *Proc. of ICCV*, 2021. 2, 3, 7, 8

[17] Jianyuan Guo, Kai Han, Han Wu, Yehui Tang, Xinghao Chen, Yunhe Wang, and Chang Xu. Cmt: Convolutional neural networks meet vision transformers. In *Proc. of CVPR*, 2022. 2

[18] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *Proc. of ECCV*, 2020. 2, 3

[19] Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the big data paradigm with compact transformers. *ArXiv preprint*, abs/2104.05704, 2021. 2

[20] Andrew Howard, Ruoming Pang, Hartwig Adam, Quoc V. Le, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, and Yukun Zhu. Searching for mobilenetv3. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1314–1324. IEEE, 2019. 3, 7

[21] Tao Huang, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Greedynasv2: Greedier search with a greedy path filter. In *Proc. of CVPR*, 2022. 7

[22] Yanping Huang, Youlong Cheng, Ankur Bapna, Orhan Firat, Dehao Chen, Mia Xu Chen, HyoukJoong Lee, Jiquan Ngiam, Quoc V. Le, Yonghui Wu, and Zhifeng Chen. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 103–112, 2019. 8

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 8

[24] Jiashi Li, Xin Xia, Wei Li, Huixia Li, Xing Wang, Xuefeng Xiao, Rui Wang, Min Zheng, and Xin Pan. Next-vit: Next generation vision transformer for efficient deployment in realistic industrial scenarios. *ArXiv preprint*, abs/2207.05501, 2022. 2

[25] Wei Li, Xing Wang, Xin Xia, Jie Wu, Xuefeng Xiao, Min Zheng, and Shiping Wen. Sepvit: Separable vision transformer. *ArXiv preprint*, abs/2203.15380, 2022. 2

[26] Yanyu Li, Geng Yuan, Yang Wen, Eric Hu, Georgios Evangelidis, Sergey Tulyakov, Yanzhi Wang, and Jian Ren. Efficientformer: Vision transformers at mobilenet speed. *ArXiv preprint*, abs/2206.01191, 2022. 2

[27] Yi-Lun Liao, Sertac Karaman, and Vivienne Sze. Searching for efficient multi-stage vision transformers. *ArXiv preprint*, abs/2109.00642, 2021. 2

[28] Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. Mcunet: Tiny deep learning on iot devices. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 2

[29] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 3

[30] Jihao Liu, Xin Huang, Guanglu Song, Hongsheng Li, and Yu Liu. Uninet: Unified architecture search with convolution, transformer, and mlp. In *Proc. of ECCV*, 2022. 3, 7

[31] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. 2021. 1, 2, 3

[32] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018. 7

[33] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *ArXiv preprint*, abs/2110.02178, 2021. 2, 7

[34] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers. *ArXiv preprint*, abs/2206.02680, 2022. 2, 7

[35] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2006. 8

[36] Junting Pan, Adrian Bulat, Fuwen Tan, Xiatian Zhu, Lukasz Dudziak, Hongsheng Li, Georgios Tzimiropoulos, and Brais Martinez. Edgevits: Competing light-weight cnns on mobile devices with vision transformers. *ArXiv preprint*, abs/2205.03436, 2022. 2, 7

[37] Namuk Park and Songkuk Kim. How do vision transformers work? In *Proc. of ICLR*, 2022. 3, 6

[38] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3498–3505. IEEE Computer Society, 2012. 8

[39] Houwen Peng, Hao Du, Hongyuan Yu, Qi Li, Jing Liao, and Jianlong Fu. Cream of the crop: Distilling prioritized paths for one-shot neural architecture search. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. 7

[40] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *Proc. of ICML*, 2018. 2

[41] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Proc. of NeurIPS*, 2021. 6

[42] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 4780–4789. AAAI Press, 2019. 3

[43] Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Jie Tan, Quoc V. Le, and Alexey Kurakin. Large-scale evolution of image classifiers. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 2902–2911. PMLR, 2017. 3

[44] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. IEEE Computer Society, 2018. 3

[45] Noam Shazeer, Zhenzhong Lan, Youlong Cheng, Nan Ding, and Le Hou. Talking-heads attention. *ArXiv preprint*, abs/2003.02436, 2020. 3

[46] Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Vitas: Vision transformer architecture search. *ArXiv preprint*, abs/2106.13700, 2021. 2, 3

[47] Xiu Su, Shan You, Jiyang Xie, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Vitas: Vision transformer architecture search. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXI*, pages 139–157. Springer, 2022. 2, 7

[48] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE Conference on Computer Vision and Pattern Recognition,*

CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 2820–2828. Computer Vision Foundation / IEEE, 2019. 7

[49] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019. 7

[50] Chen Tang, Haoyu Zhai, Kai Ouyang, Zhi Wang, Yifei Zhu, and Wenwu Zhu. Arbitrary bit-width network: A joint layer-wise quantization and adaptive inference approach. In *Proc. of ACM MM*, 2022. 5

[51] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers and distillation through attention. In *Proc. of ICML*, 2021. 1

[52] Shakti N Wadekar and Abhishek Chaurasia. Mobilevitv3: Mobile-friendly vision transformer with simple and effective fusion of local, global and input features. *ArXiv preprint*, abs/2209.15159, 2022. 2, 7

[53] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, Peter Vajda, and Joseph E. Gonzalez. Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 12962–12971. IEEE, 2020. 7

[54] Dilin Wang, Chengyue Gong, Meng Li, Qiang Liu, and Vikas Chandra. Alphanet: Improved training of supernets with alpha-divergence. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10760–10771. PMLR, 2021. 3, 6

[55] Dilin Wang, Meng Li, Chengyue Gong, and Vikas Chandra. Attentivenas: Improving neural architecture search via attentive sampling. In *Proc. of CVPR*, 2021. 4

[56] Xudong Wang, Li Lyna Zhang, Yang Wang, and Mao Yang. Towards efficient vision transformer inference: A first study of transformers on mobile devices. In *Proceedings of the 23rd Annual International Workshop on Mobile Computing Systems and Applications*, 2022. 1

[57] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *Proc. of ICCV*, 2021. 2

[58] Jin Xu, Xu Tan, Kaitao Song, Renqian Luo, Yichong Leng, Tao Qin, Tie-Yan Liu, and Jian Li. Analyzing and mitigating interference in neural architecture search. In *Proc. of ICML*, 2022. 7

[59] Jiahui Yu, Pengchong Jin, Hanxiao Liu, Gabriel Bender, Pieter-Jan Kindermans, Mingxing Tan, Thomas Huang, Xiaodan Song, Ruoming Pang, and Quoc Le. Bignas: Scaling up neural architecture search with big single-stage models. In *Proc. of ECCV*, 2020. 1, 2, 3, 4, 7

[60] Kaicheng Yu, Christian Sciuto, Martin Jaggi, Claudiu Musat, and Mathieu Salzmann. Evaluating the search phase of neural architecture search. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 2

[61] Li Lyna Zhang, Shihao Han, Jianyu Wei, Ningxin Zheng, Ting Cao, Yuqing Yang, and Yunxin Liu. nn-meter: Towards accurate latency prediction of deep-learning model inference on diverse edge devices. In *The 19th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys 2021)*, 2021. 6

[62] Li Lyna Zhang, Yuqing Yang, Yuhang Jiang, Wenwu Zhu, and Yunxin Liu. Fast hardware-aware neural architecture search. In *Proc. of CVPR*, 2020. 2

[63] Xinbang Zhang, Zehao Huang, Naiyan Wang, Shiming Xiang, and Chunhong Pan. You only search once: Single shot neural architecture search via direct sparse optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 2

[64] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1

[65] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. 3

[66] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8697–8710. IEEE Computer Society, 2018. 3