ViTAS: Vision Transformer Architecture Search

Xiu Su[1], Shan You[2,3] *, Jiyang Xie[4], Mingkai Zheng[1], Fei Wang[5], and Chen Qian[2] Changshui Zhang[3] Xiaogang Wang[2,6] Chang Xu[1]

[1] School of Computer Science, Faculty of Engineering, The University of Sydney.
xisu5992@uni.sydney.edu.au, c.xu@sydney.edu.au
[2] SenseTime Research. {youshan,qianchen}@sensetime.com
[3] Department of Automation, THUAI, BNRist, Tsinghua University
zcs@mail.tsinghua.edu.cn
[4] Beijing University of Posts and Telecommunications. xiejiyang2013@bupt.edu.cn
[5] University of Science and Technology of China. wangfei91@mail.ustc.edu.cn
[6] The Chinese University of Hong Kong. xgwang@ee.cuhk.edu.hk

Abstract. Vision transformers (ViTs) inherited the success of NLP but their structures have not been sufficiently investigated and optimized for visual tasks. One of the simplest solutions is to directly search the optimal one via the widely used neural architecture search (NAS) in CNNs. However, we empirically find this straightforward adaptation would encounter catastrophic failures and be frustratingly unstable for the training of superformer. In this paper, we argue that since ViTs mainly operate on token embeddings with little inductive bias, imbalance of channels for different architectures would worsen the weight-sharing assumption and cause the training instability as a result. Therefore, we develop a new cyclic weight-sharing mechanism for token embeddings of the ViTs, which enables each channel could more evenly contribute to all candidate architectures. Besides, we also propose identity shifting to alleviate the many-to-one issue in superformer and leverage weak augmentation and regularization techniques for more steady training empirically. Based on these, our proposed method, ViTAS, has achieved significant superiority in both DeiT- and Twins-based ViTs. For example, with only 1.4G FLOPs budget, our searched architecture achieves 3.3% higher accuracy than the baseline DeiT on ImageNet-1k dataset. With 3.0G FLOPs, our results achieve 82.0% accuracy on ImageNet-1k, and 45.9% mAP on COCO2017, which is 2.4% superior than other ViTs.

Keywords: Vision transformer (ViT), nerual architecture search (NAS), cyclic weight sharing mechanism, identity shifting, weak augmentation

1 Introduction

Transformer, as a self-attention characterized neural network, has been widely leveraged for natural language processing (NLP) tasks [9,31,32,1]. Amazingly, recent breakthrough of vision transformers (ViTs) [11,41] further revealed the huge potential of transformers in computer vision (CV) tasks [62,64,65,49,47,48,38,52]. With no use of inductive biases, self-attention layers in the transformer introduce a global receptive

* Corresponding author.

field, which conveys refresh solutions to process vision data. Following ViTs, there have been quite a few works on vision transformers for a variety of tasks, such as image recognition [20,12,7,59,54,63], object detection [68,25], and semantic segmentation [25].

Despite the remarkable achievements of ViTs, the design of their architectures is still rarely investigated. Current ViTs simply split an image into a sequence of patches (*i.e.*, tokens) and stack transformer blocks as NLP tasks. Nevertheless, this vanilla protocol does not necessarily ensure the optimality for vision tasks. Stacking manner and intrinsic structure of the blocks need to be further analyzed and determined, such as patch size of input, head number in multihead self-attention (MHSA), output dimensions of parametric layers, operation type, and depth of the whole model. Therefore, we raise questions that *What makes a better vision transformer? How can we obtain it?* Inspired by the success of one-shot neural architecture search (NAS) in ConvNets (CNNs), our intuition is also to directly search for an optimal architecture for ViTs, which in turn gives us insight about designing more promising ViTs.

Unlike the sliding convolutions of CNNs, ViTs project the patches into a sequence of token embeddings, and the features are extracted sequentially. In this way, how to specify an appropriate configuration (dimension) for token embeddings of all layers play an important role for the architecture of ViTs [51]. To search [43] for the optimal token embedding dimension, recent work [4,21] simply borrow the ordinal weight sharing [14,60] in CNNs for the superformer (a.k.a. supernet in CNNs) to accommodate different token dimension. However, this ordinal mechanism would inevitably introduce imbalance among channels during training, causing the superformer cannot evaluate each token dimension well and induces sub-optimal architectures consequently. Though recent bilateral mechanism [37] was proposed to handle this issue, the training cost has to be doubled yet the imbalance of channels still exist to some extent.

In this paper, we propose a novel *cyclic* weight sharing mechanism for superformer to embody various token embedding dimensions of all layers. Concretely, we encourage balanced *training fairness* and *influence uniformity* for each channel in the superformer. With these two conditions, the cyclic rule could be learned as an index mapping to indicate each dimension of token embeddings (see Figure 1), so that each could be more evenly evaluated. Besides, since the cyclic rule is a single-pass mapping, computation cost of training the superformer is similar to that of a ordinal [60,4] one.

Based on the customized cyclic manner, we propose a corresponding NAS method for ViTs dubbed vision transformer architecure search (ViTAS). However, we empirically observe that the training of superformer tends to be frustratingly instable. We argue that the space size of ViTs are way too huger (even 1.1×10^{54}), and propose to calibrate the space with an identity shifting technique. Besides, we find that strong augmentation and regularization are critic to further stabilize the superformer training. Extensive experimental results have shown the superiority of our ViTAS.

2 Related Work

Vision Transformer. ViT was first proposed by Dosovitskiy et al. [11] to extend the applications of transformers into computer vision fields by cascading manually designed

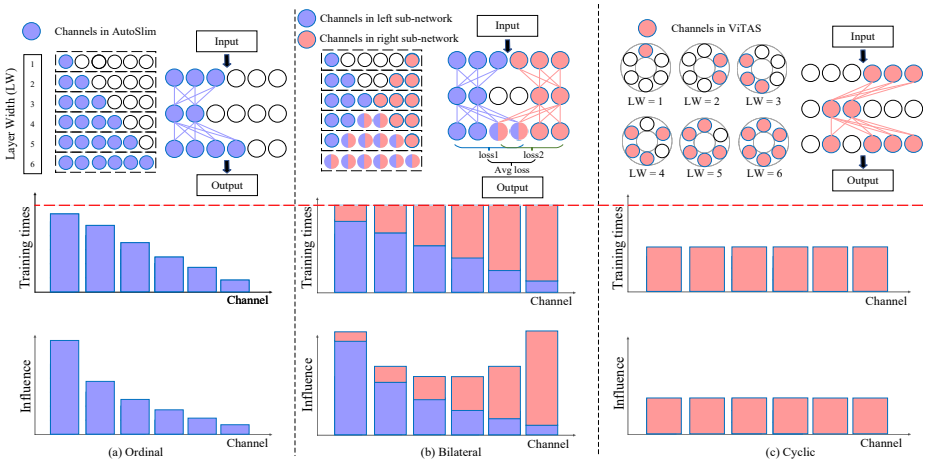


Fig. 1: Comparison between (a) ordinal [60,4], (b) bilateral [37] and (c) our cyclic forms with a toy example of six groups of channels. In (a), channels cannot be fairly trained in terms of training times and influence. Channels in (b) need double training cost compared to the others and also unevenly distributed influence. Then in (c), the proposed cyclic pattern overcomes the defects of (a) and (b) and achieves fairly training of channels w.r.t. training times and influence while maintaining half training cost than the bilateral form.

multilayer perceptrons (MLPs) and MHSA modules. Touvron et al. [41] introduced a teacher-student strategy and a distillation token into the ViT, namely data-efficient image transformers (DeiT). Recently, other variants of ViT were proposed and all introduced inductive bias and prior knowledge to extract local information for better feature extraction. Tokens-to-Token (T2T) ViT [61] added a layer-wise T2T transformation and a deep-narrow backbone to overcome limitations of local structure modeling. Then, Han et al. [15] proposed to model both patch- and pixel-level representations by transformer-in-transformer (TNT). Swin Transformer [25] generated various patch scales by shifted windows for better representing highly changeable visual elements. Wu et al. [45] reintroduced convolutions into the ViT, namely convolutional vision transformer (CvT). Pyramid vision transformer (PVT) [44] trained on dense partitions of the image to achieve high output resolution and used a progressive shrinking pyramid to reduce computations of large feature maps. Another Twins ViT framework [5] was proposed, which introduced spatially separable self-attention (SSSA) to replace the less efficient global sub-sampled attention in PVT. All the aforementioned transformer structures were manually designed according to expert experience.

One-shot NAS Method. Differentiable architecture search (DARTS) [24,56,57,19] first formulated the NAS task in a differentiable manner based on the continuous relaxation. In contrast, single path one-shot (SPOS) framework [53,35,58] adopted an explicit path sampler to construct a simplified supernet, such as uniform sampler [14,38], greedy sampler [58,18,53] and Monte-Carlo tree sampler [34]. Some work also at-

Table 1: Macro transformer space for the ViTAS of Twins-based architecture."TBS" indicates that layer type is searched from parametric operation and identity operation for depth search. "Embedding" represents the patch embedding layer. "Max_a" and "Max_m" indicates the max dimension of attention layer ("Max_a" also used for patch embedding layer) and mlp layer, respectively. "Ratio" means the reduction ratio from the "Max Output Dim". A larger "Ratio" indicates a larger dimension.

Number	OP	Type	Patch size / #Heads	Max_a	Max_m	Ratio
1	False	Embeding	4	128	-	$\{i/10\}_{i=1}^{10}$
4	TBS	Local Global	$\{2,4,8,16\}$	480	512	$\{i/10\}_{i=1}^{10}$
1	False	Embeding	2	256	-	$\{i/10\}_{i=1}^{10}$
4	TBS	Local Global	$\{2,4,8,16\}$	960	1024	$\{i/10\}_{i=1}^{10}$
1	False	Embeding	2	512	-	$\{i/10\}_{i=1}^{10}$
12	TBS	Local Global	$\{2,4,8,16\}$	1920	2048	$\{i/10\}_{i=1}^{10}$
1	False	Embeding	2	1024	-	$\{i/10\}_{i=1}^{10}$
6	TBS	Local Global	$\{2,4,8,16\}$	3840	4096	$\{i/10\}_{i=1}^{10}$

tempted to investigate the channel dimension by direct searching [37,36] or pruning from pretrained models [26,39]. As for ViTs, AutoFormer [4] first adopted the one-shot NAS framework for the ViT based architecture search. BossNAS [21] implemented the search with an self-supervised training scheme and leveraged a hybrid CNN-transformer search space for boosting the performance.

3 Revisiting One-shot NAS towards Transformer Space

One-shot NAS & dimension search. Towards the search of a decent architecture $\boldsymbol{\alpha} \in \mathcal{A}$ from a huge transformer space $\mathcal{A}$ (*i.e.*, transformer space), a weight sharing strategy is commonly leveraged to avoid exhausted path training from scratch. For a superformer $\mathcal{N}$ with weights $\mathcal{W}$, each path $\boldsymbol{\alpha}$ inherits its weights from $\mathcal{W}$. The one-shot NAS is thus formulated as a two-stage optimization problem, *i.e.*, superformer training and then architecture searching. Base on the above settings, many researchers leveraged dimension search algorithms, *e.g.*, AutoSlim [60,4] and BCNet [37], to perform the search of the dimensions for fine grained architectures. We define $\mathcal{C}$ as the set of candidate dimensions for a certain operation, where $\boldsymbol{c} \in \mathcal{C}$ indicates the dimensions within $\boldsymbol{\alpha}$. Thus, the optimization function is as

$$W^*_{\mathcal{A},\mathcal{C}} = \underset{W_{\mathcal{A},\mathcal{C}}}{\operatorname{argmin}} \, loss_{\text{train}} \left(\mathcal{N}(\mathcal{A},\mathcal{C},W_{\mathcal{A},\mathcal{C}})\right), \tag{1}$$

$$\boldsymbol{\alpha}^*, \boldsymbol{c}^* = \underset{(\boldsymbol{\alpha},\boldsymbol{c})\in(\mathcal{A},\mathcal{C})}{\operatorname{argmax}} \, Acc_{\text{val}} \left(\mathcal{N}(\boldsymbol{\alpha},\boldsymbol{c},W^*_{\boldsymbol{\alpha},\boldsymbol{c}})\right), \tag{2}$$

$$\text{s.t. FLOPs}(\mathcal{N}(\boldsymbol{\alpha},\boldsymbol{c},W^*_{\boldsymbol{\alpha},\boldsymbol{c}})) \leq f,$$

Table 2: Macro transformer space for the ViTAS of DeiT-based architecture."TBS" indicates that layer type is searched from vanilla ViT block or identity operation for depth search. "Ratio" means the reduction ratio from the "Max Dim". A larger "Ratio" means a larger dimension.

Number	OP	Type	Patch size / #Heads	Max Dim	Ratio
1	False	Linear	$\{14, 16, 32\}$	384	$\{i/10\}_{i=1}^{10}$
16	TBS	MHSA	$\{3, 6, 12, 16\}$	1440	$\{i/10\}_{i=1}^{10}$
		MLP	-	1440	$\{i/10\}_{i=1}^{10}$

where $loss_{\text{train}}$ is training loss, Acc_{val} is validation accuracy, $W^*_{\mathcal{A},\mathcal{C}}$ is a set of trained weights, $\boldsymbol{\alpha}^*$ is the searched optimal architecture, and f is resource budget. Following the one-shot framework, the superformer is trained by uniformly sampling different $(\boldsymbol{\alpha}, \boldsymbol{c})$ from $(\mathcal{A}, \mathcal{C})$, and then we search the optimal architecture $(\boldsymbol{\alpha}^*, \boldsymbol{c}^*)$ according to W^*. After these, the selected $\boldsymbol{\alpha}^*$ will be retrained for evaluation.

Towards Transformer Space. To explore the possibility of the optimal ViT architecture in the arch-level, we incorporate all the essential elements in our transformer space, including *head number*, *patch size*, *operation type*, *output dimension of each layer*, and *depth of the architectures*, as shown in Table 1[7] and Table 2[8]. More details of transformer space is elaborated in the Section A.2 of Appendix. With the Twins-small based transformer space in Table 1 as an example, the size of transformer space amounts to 1.1×10^{54} and the FLOPs (parameters) ranges from 0.02G (0.16M) to 11.2G (86.1M). Similarly in Table 2, with the DeiT-small transformer space, the size of space amounts to 5.4×10^{34} and the FLOPs (parameters) ranges from 0.1G (0.5M) to 20.0G (97.5M).

4 Cyclic Channels for Token Embeddings

Previous work [60,4] proposed the ordinal weight sharing paradigm, which is widely leveraged in many CNN and Transformer NAS papers [4,42,55]. Concretely, as illustrated in Figure 1(a), to search for a dimension i at a layer with maximum of l channels, the ordinal pattern assigns the left i channels in the superformer to indicate the corresponding architecture as

$$\boldsymbol{a}_A(i) = [1 : i],\ i \leq l, \tag{3}$$

where $\boldsymbol{a}_A(i)$ means the selected i channels from the left (smaller-index) side.

However, this channel configuration imposes a strong constraint on the channels and leads to imbalanced training for each channel in the superformer. As in Figure 1(a), with the ordinal pattern, channels that are close to the left side are used in both large and

[7] In the superformer, Max_a indicates the output of the first fully connected (FC) layer, which should be able to be divided by all "ratios" and "Heads", *i.e.*, $\text{Max}_\text{a}|(Ratio \times Heads), \forall Ratio, Heads$. Therefore, we select least common multiple of "ratios" and "Heads" for Max_a

[8] In the superformer, "Max Dim" indicates the output dimensions of both attention and MLP blocks.

small dimension. Since different dimensions are uniformly sampled during searching, the training times $\mathcal{C}_A(i)$ of the i-th channel used in all dimensions with the ordinal pattern can be represented as

$$\mathcal{C}_A(i) = l - i + 1. \tag{4}$$

Therefore, channels closer to the left side will gain more times of training, which induces evaluation bias among different channels and leads to sub-optimal searching results.

To remove the evaluation bias among channels of the superformer, we introduce a condition for constructing a mapping for channels:

Theorem 1 (training fairness). *Each channel should obtain same training times for fairer training of superformer.*

With the aims of **Theorem** 1 and keep the same computation cost as AutoSlim [60,4] (*i.e.*, ordinal pattern), we introduce indicator matrix β with $\beta_{i,j} \in \{0, 1\}$ (one means using the channel) to represent whether channel i being used in dimension j. Two conditions need to be satisfied: (1) for each row β_i, which is the training times of the channel i in each dimension, the sum of it should be equal with that of all the other channels, and (2) for each column β_j, which demonstrates the training times of channels in dimension j, the sum of it should be the dimension of itself. Finally, the constraints of β can be represented as follows

$$\sum_j \beta_{i,j} = (1 + l)/2, \ \forall i, \tag{5}$$

$$\sum_i \beta_{i,j} = j, \ \forall j. \tag{6}$$

Infinite solutions can be solved under aforementioned constraints only. Here, the bilateral pattern in BCNet [37] is a special case of the aforementioned settings with double training times as $l + 1$.

Although forcing the channels to be trained for same training times can boost the fairness, constructing a path only constrained by condition 1 cannot emerge the actual performance of the path due to the difference of training saturation between one-shot-based sampling and training from scratch [27].

This means that in order to more precisely rank various paths, we need to mimic the process of the latter and balance the influence of each channels.

Theorem 2 (influence uniformity). *Each channel needs to have the same sum of influence among all its related dimensions for training.*

Concretely, we should carefully design the weight sharing mechanism based on **Theorem 2**. Here, we define $\psi_{i,j}$ to indicate the influence of channel i in dimension j. For each channel pair, we have

$$\begin{cases} \psi_{i_1,j} = \psi_{i_2,j}, \forall i_1, i_2 \\ \psi_{i,j_1} \geq \psi_{i,j_2}, \forall j_1 \leq j_2 \end{cases}. \tag{7}$$

Considering that transformer architectures are mainly consist of full-connected (FC) layers. Specifically, for an FC layer with j input channels $x_i, i = 1, \cdots, j$ and a certain output channel $y = \sum_{i=1}^{j} y_i, y_i = w_i x_i$ with parameters $w_i, i = 1, \cdots, j$, we can obtain the gradient of w_i as

$$\bigtriangledown w_i = \frac{\partial loss_{\text{train}}}{\partial y} \cdot \frac{\partial y}{\partial y_i} \cdot \frac{\partial y_i}{\partial x_i}. \tag{8}$$

Meanwhile, for a dimension sampled from the FC layer with one random input channel x_i, the gradient of w_i here can be represented as

$$(\bigtriangledown w_i)' = \frac{\partial loss_{\text{train}}}{\partial y'} \cdot \frac{\partial y'}{\partial y_i} \cdot \frac{\partial y_i}{\partial x_i}, \tag{9}$$

where $y' = y_i$. Therefore, in the former case, the influence $\psi_{i,j}$ of the i-th channel can be defined as the contribution of the channel to its gradient as

$$\psi_{i,j} = \frac{(\bigtriangledown w_i)'}{\bigtriangledown w_i} = \frac{\frac{\partial y'}{\partial y_i}}{\frac{\partial y}{\partial y_i}}, \tag{10}$$

assuming $\frac{\partial loss_{\text{train}}}{\partial y} = \frac{\partial loss_{\text{train}}}{\partial y'}$. We can also assume $y_i \approx y_{i'}, i \neq i'$, as the distributions of w_i or x_i can be similar to each i, respectively, when randomly sampling the dimensions in each batch. In this case, we can obtain $y \approx j \times y_i$, and $\psi_{i,j} = \frac{1}{j}$ for Eq. (7) and (10).

Note that channel i may be shared w.r.t.different dimensions. To keep all the channels being treated equally, any two channels i_1 and i_2 should have the same influence among all the dimensions, *i.e.*,

$$\sum_j \beta_{i_1,j} \psi_{i_1,j} = \sum_j \beta_{i_2,j} \psi_{i_2,j}, \ \forall i_1, i_2. \tag{11}$$

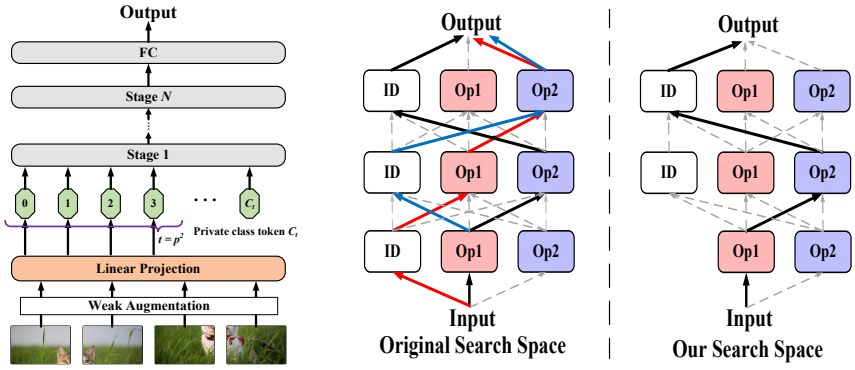
Optimization of cyclic mapping. Combining Eq. (5) $\sim$ Eq. (11), we can obtain the specialized weight sharing paradigm for the cyclic superformer. In practice, since $\frac{1+l}{2}$ may not be an integer, Eq. (5) may not be completely satisfied. Thus, for any two channels i_1 and i_2, we can relax the constraint in Eq. (5) by

$$\left| \sum_j \beta_{i_1,j} - \sum_j \beta_{i_2,j} \right| \leq 1, \ \forall i_1, i_2. \tag{12}$$

To facilitate the search of the optimal weight sharing paradigm, we should make sure all the channels being fairly trained with almost the same influence among all the dimensions. Therefore, we can update Eq. (11) to an objective as

$$\min_{\beta} \sum_{i_1,i_2} \left(\beta_{i_1,j} \psi_{i_1,j} - \beta_{i_2,j} \psi_{i_2,j} \right)^2, \tag{13}$$

The overall problem is thus a QCQP (quadratically constrained quadratic program), which can be efficiently solved by many off-the-shelf solvers [10,29]. We have presented detailed experimental settings and simulations of β and $\psi_{i,j}$ in Section A.15 of Appendix.



(a) Class token in ViTAS. (b) Identity shifting.

Fig. 2: Private class token and identity shifting search space in ViTAS. (a) For transformer architectures with class token settings, with different patch sizes p, we assign one independent class tokens for each and obtain $p \times p$ patches and a private class token under one patch size setting. (b) Comparison between currently mainstream ID search strategy (left) and ours ID oriented method (right). From left, a group of paths in the superformer, *i.e.*, **red**, **blue** and **black** paths, corresponds to one same architecture, which may performs differently. This can hinder the search for decent architectures. While for our method, each architecture corresponds to only one path in the superformer, which reduces the redundancy in it and boosts the search performance.

5 Further Stabilizing the Training of Superformer

Training the superformer is for fair estimation of each architecture's performance, which is essential for the next optimal architecture searching stage. Here, we argue that superformer requires an efficient and simple transformer space and training recipe for boosting the search. For the transformer space of ViTAS, we propose the identity shifting strategy to solve the many-to-one issue as in Figure 2b. Besides, for architectures with class tokens, *i.e.*DeiT [41], we introduce private class token w.r.t. each patch size to cater for different paths. For the training recipe of ViTAS, we underline that weak augmentation & regularization rather than complex and tricky ones [41,5] can prevent the search from unsteady.

Identity shifting. Given a pre-defined NAS transformer space, as in Figure 2b, identity (ID) operation serves as the significant part and has a large effect on the searched results for three reasons: 1) it defines the depth of the searched architecture, 2) compared to other operations, the non-parametric ID is much more different with other parametric operations, which will involve in a higher variance on paths, and 3) stacking manner and intrinsic structure of a transformer architecture within each stage lead to complicatedly many-to-one correspondence between architectures in the superformer and transformer space. These introduce a huge ambiguous for NAS. Here, we propose to search the operations with identity shifting strategy, as depicted in the right side of Figure 2b. In each stage, we remove the ambiguous between transformer space and su-

Table 3: Searched Twins-based ViT architectures w.r.t. different FLOPs and GPU throughput on ImageNet-1k. We abbreviate the name of tiny, short, base, and large for T, S, B, and L, respectively. $\star$ indicates that the re-implementation results of important baseline methods with our recipe. Our results are highlighted in bold.

Method	FLOPs (G)	Throughput (image/s)	Params (M)	Top-1 (%)	Top-5 (%)
ResNet-18 [17]	1.8	4458.4	12	69.8	89.1
DeiT-T* [41]	1.3	2728.5	5	72.3	91.4
Twins-T* [5]	1.4	1580.7	11.5	77.8	94.1
AutoFormer-T* [4]	1.3	3055.4	5.7	74.7	91.9
ViTAS-Twins-T	1.4	1686.3	13.8	**79.4**	**94.8**
DeiT-S* [41]	4.6	437.0	22.1	79.9	95.0
PVT-S [44]	3.8	820	24.5	79.8	-
Twins-SVT-S* [5]	2.9	1059	24	81.6	95.9
AutoFormer-S* [4]	5.1	1231.7	22.9	79.8	95.0
BossNet-T0 [21]	5.7	-	-	81.6	-
Twins-PCPVT-S* [5]	3.8	815	24.1	81.2	95.6
Swin-T* [25]	4.5	766	29	81.2	95.5
ViTAS-Twins-S	3.0	958.6	30.5	**82.0**	**95.7**
T2T-ViT$_t$-19 [61]	8.9	-	39.2	81.4	-
BoTNet-S1-59 [33]	7.3	-	33.5	81.7	-
BossNet-T1 [21]	7.9	-	-	82.2	-
Twins-PCPVT-B [5]	6.7	525	43.8	82.7	-
Swin-S* [25]	8.7	444	50	83.0	96.2
Twins-SVT-B* [5]	8.6	469	56	83.2	96.3
ViTAS-Twins-B	8.8	362.7	66.0	**83.5**	**96.5**
DeiT-B* [41]	17.6	292	86.6	81.8	95.7
TNT-B [15]	14.1	-	66	82.8	-
CrossViT-B [2]	21.2	-	104.7	82.2	-
ViTAS-Twins-L	16.1	260.7	124.8	**84.0**	**96.9**

performer by sampling the number of ID and arrange them at the deeper layers of the stage in order to remove the redundancy in superformer. Typically, with three operations (including ID) and twelve searched layers, the transformer space of operations can be reduced from 3^{12} to $2^{13}-1$.

Private class token. Notably, pure vision transformer architectures, *e.g.*, DeiT [41], usually introduce a trainable vector named class token for the output classification. The class token is appended to the patch tokens before the first layer and then go through the transformer blocks for the prediction task. These class tokens often take a small size, which changes with the pre-defined patch size P (*i.e.*, $\frac{H\times W}{P\times P}$), and performs significant in performance. Towards these attributes, we propose to privatize the class token for each P. As shown in Figure 2a, for different patch sizes, we assign private ones for each. In this way, the affect between class tokens can be avoided with only negligible computation cost or memory cost introduced. Ablations of the private class token are presented in Section A.6 of Appendix.

Weak augmentation & regularization. We explore the superformer training strategy of the ViTAS, including data augmentation and regularization. We conducted the evaluations with Twins-based architecture and 1.4G FLOPs budget on ImageNet-1k dataset.

Table 4: Ablation studies of training recipe of superformer of the ViTAS. "✓"/"✗" indicates that we used/not used the corresponding method. We implemented the search on the ImageNet-1k with 1.4G budget of Twins. We report the top-1 accuracy of the best architectures in both ViTAS (*i.e.*, searching) and retraining. "RD": Rand-Augment. "MP": Mixup. "CM": CutMix. "CJ": Color Jitter. "Era": Erasing. "SD": Stoch Depth. "RA": Repeated Augmentation. "WD": Weight Decay. Best results are in bold.

	Data augmentation				Regularization				ViTAS	Retraining
#	RD	MP	CM	CJ	Era	SD	RA	WD	Acc(%)	Acc(%)
0	✓	✓	✓	✓	✓	✓	✓	✓	59.4%	77.9%
1	✓	✓	✓	✓	✓	✗	✓	✓	61.2%	78.0%
2	✓	✓	✓	✓	✓	✗	✗	✓	61.5%	78.2%
3	✓	✗	✗	✓	✓	✗	✗	✓	62.3%	78.6%
4	✗	✗	✗	✓	✓	✗	✗	✓	64.9%	78.8%
7	✗	✗	✗	✗	✓	✗	✗	✓	65.6%	78.9%
8	✗	✗	✗	✗	✗	✗	✗	✓	66.1%	79.1%
9	✗	✗	✗	✗	✗	✗	✗	✗	**67.7%**	**79.4%**

Table 5: Training recipe of the ViTAS with parameter settings. BS: batch size, LR: learning rate, WD: weight decay. We will conduct the ViTAS according to the following recipe in experiments.

Epochs	BS	Optimizer	LR	LR decay	Warmup
300	1024	AdamW	0.001	cosine	5

Compared with one single ViT, the superformer training is much more difficult to converge, which needs a simply yet effective training strategy.

From Table 4, as group 0, the superformer performs badly with the default training strategy [41,5]. To facility the search, we first removed the stochastic depth, since our identity search performs the similar effect in the superformer training. Then, we gradually dropped other data augmentation and/or regularization and find that a weak augmentation can largely promote retraining accuracy with searching a better architecture. Table 5 presents the ViTAS training recipe for our experiments. We provide a detailed analysis of weak augmentation in Section A.14 of Appendix.

6 Experimental Results

We perform the ViTAS on the challenging ImageNet-1k dataset [8] for image classification, and COCO2017 [13] and ADE20k [67] for object detection, instance segmentation, and semantic segmentation. To promote the search, we randomly sample 50K images from the training set as the local validation set and the rest images are leveraged for training. All experiments are implemented with PyTorch [30] and trained on V100 GPUs. Please find detailed experimental settings in Section A.1 and A.3 of Appendix.

Table 6: Searched ViT architectures that do not involve inductive bias w.r.t. different FLOPs and GPU throughput on ImageNet-1k. $\star$ indicates that the re-implementation results of important baseline methods with our recipe. Our results are in bold.

Method	FLOPs (G)	Throughput (image/s)	Params (M)	Top-1 (%)	Top-5 (%)
ResNet-18 [17]	1.8	4458.4	12	69.8	89.1
DeiT-T* [41]	1.3	2728.5	5	72.3	91.4
AutoFormer-T* [4]	1.3	2955.4	5.7	74.7	91.9
ViTAS-DeiT-A	1.4	2831.1	6.6	**75.6**	**92.5**
ResNet-50[17]	4.1	1226.1	25	76.2	91.4
DeiT-S* [41]	4.6	940.4	22	79.9	95.0
AutoFormer-S* [4]	5.1	1231.7	22.9	79.8	95.0
ViTAS-DeiT-B	4.9	1189.4	2.3	**80.2**	**95.1**

6.1 Efficient search of ViTAS on ImageNet-1k

In Table 3, we compare our results with recent precedent ViT architectures. To evaluate our methods with other existing algorithms, we based on the Twins transformer space[9] and present the search results with both FLOPs and GPU throughput. With different FLOPs budgets, The search ones (ViTAS-Twins-T/S/B/L) can outperform the referred transformers. For example, ViTAS-Twins-T can improve Top-1 accuracy by 1.6%, compared with Twins-T. For larger architectures, the search ones can moderately surpass all the corresponding referred ones as well, respectively.

In addition, we also searched the optimal architectures based on pure ViT and DeiT space, shown in Table 6. With only 1.4G FLOPs and similar GPU throughput, our searched ViTAS-DeiT-A model achieves 75.6% on Top-1 accuracy and is 3.4% superior than DeiT-T, which indicates the effectiveness of our proposed ViTAS method. Furthermore, with 4.9G FLOPs budget, our ViTAS-DeiT-B model also achieves superior performance of 80.2% on Top-1 accuracy with 0.3% surpassing the DeiT-S.

6.2 Transferability of ViTAS with Semantic Segmentation on ADE20K

In addition to search the optimal ViT architectures on the ImageNet-1k, we evaluated the generalization ability of the ViTAS by transferring the searched architectures to other tasks. With the same recipe as Twins [5], we fintuned on ADE20k [67] by using our ImageNet-pretrained models as backbones for semantic segmentation, shown in Table 8. Under different FLOPs budgets, our models can obtain significant performance improvement as $2\% \sim 4\%$ on mIoU, compared with corresponding referred methods. For example, with semantic FPN [5] method as baseline, ViTAS-Twins-B surpasses the second best one, Twins-SVT-B, by more than 4% on mIoU.

[9] We constructed the ViTAS-Twins-T transformer space from Twins-S similar to Table 1, and the Twins-T was uniformly scaled from Twins-S.

Table 7: Object detection and instance segmentation performance with searched backbones on the COCO2017 dataset with Mask R-CNN framework and RatinaNet framework. We followed the same training and evaluation setting as [5]. "FLOPs" and "Param" are in giga and million, respectively. $\star$ indicates the re-implementation results of important baseline methods with our recipe. Our results are highlighted in bold.

Backbone	Mask R-CNN 1× [16]								RetinaNet 1× [22]							
	FLOPs	Param	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	FLOPs	Param	AP^b	AP^b_{50}	AP^b_{75}	AP_S	AP_M	AP_L
ResNet50 [17]	174	44.2	38.0	58.6	41.4	34.4	55.1	35.7	111	37.7	36.3	55.3	38.6	19.3	40.0	48.8
PVT-Small [44]	178	44.1	40.4	62.9	43.8	37.8	60.1	40.3	118	34.2	40.4	61.3	43.0	25.0	42.9	55.7
Twins-PCPVT-S [5]	178	44.3	42.9	65.8	47.1	40.0	62.7	42.9	118	34.4	43.0	64.1	46.0	27.5	46.3	57.3
Swin-T [25]	177	47.8	42.2	64.6	46.2	39.1	61.6	42.0	118	38.5	41.5	62.1	44.2	25.1	44.9	55.5
Twins-SVT-S* [5]	164	44.0	43.5	66.0	47.8	40.1	62.9	43.1	104	34.3	42.2	63.3	44.9	26.4	45.6	57.0
ViTAS-Twins-S	168	44.2	**45.9**	**67.8**	**50.3**	**41.5**	**64.7**	**45.0**	108	41.3	**44.4**	**65.3**	**47.6**	**27.5**	**48.3**	**60.0**
ResNet101 [17]	210	63.2	40.4	61.1	44.2	36.4	57.7	38.8	149	56.7	38.5	57.8	41.2	21.4	42.6	51.1
ResNeXt101 [50]	212	62.8	41.9	62.5	45.9	37.5	59.4	40.2	151	56.4	39.9	59.6	42.7	22.3	44.2	52.5
PVT-Medium [5]	211	63.9	42.0	64.4	45.6	39.0	61.6	42.1	151	53.9	41.9	63.1	44.3	25.0	44.9	57.6
Twins-PCPVT-B [5]	211	64.0	44.6	66.7	48.9	40.9	63.8	44.2	151	54.1	44.3	65.6	47.3	27.9	47.9	59.6
Swin-S [25]	222	69.1	44.8	66.6	48.9	40.9	63.4	44.2	162	59.8	44.5	65.7	47.5	27.4	48.0	59.9
Twins-SVT-B* [5]	224	76.3	45.5	67.4	50.0	41.4	64.5	44.5	163	67.0	44.4	65.6	47.4	28.5	47.9	59.5
ViTAS-Twins-B	227	85.4	**47.6**	**69.2**	**52.2**	**42.9**	**66.3**	**46.5**	167	76.2	**46.0**	**66.7**	**49.6**	**29.1**	**50.2**	**62.0**
Twins-SVT-L* [5]	292	119.7	45.9	67.9	49.9	41.6	65.0	45.0	232	110.9	45.2	66.6	48.4	29.0	48.6	60.9
ViTAS-Twins-L	301	144.1	**48.2**	**69.9**	**52.9**	**43.3**	**66.9**	**46.7**	246	135.5	**47.0**	**67.8**	**50.3**	**29.6**	**50.9**	**62.4**

6.3 Transferability to Object Detection and Instance Segmentation

With the same recipe as Twins [5], we undertook both object detection and instance segmentation on COCO2017 [23] by using our ImageNet-pretrained models as backbones, respectively. In Table 7, with mask R-CNN and RetinaNet as baseline, we achieve state-of-the-art performance with remarkably improvement on each AP metrics. The aforementioned experimental results in both of the tasks can demonstrate the effectiveness of ViTAS.

6.4 Ablation Studies

Effect of ViTAS as a superformer. To validate the effectiveness of our proposed ViTAS, as in Table 9, we implemented the search with 1.4G FLOPs budget and Twins transformer space on ImageNet-1k dataset. Our baseline superformers are AutoFormer [60,4] and BCNet [37] that adopt ordinal or bilateral weight sharing mechanism, respectively, to evaluate a sampled architecture. With all settings in our paper, our cyclic pattern (79.4%) can enjoy a gain of 0.9% or 1.3% on Top-1 accuracy compare to bilateral (78.5%) or ordinal (78.1%) pattern, respectively. In addition, when only searching with the cyclic weight pattern and without additional strategies, our method (77.9%) can still attain 0.3% or 0.7% performance gain compare to baseline methods of bilateral (77.6%) and ordinal (77.2%) weight sharing mechanism. We also conduct the ablations of ViTAS w.r.t. DeiT seach space in section A.7 of Appendix.

Comparison of AutoFormer [60,4], BCNet [37], and ViTAS w.r.t weight sharing paradigm of superformer training. AutoFormer, BCNet, and ViTAS adopt the ordinal, bilaterally, and cyclic weight sharing paradigm, respectively. As in Figure 3a, we depict the average of training loss in each epoch w.r.t. three weight sharing mechanisms. In general, two obvious phenomena can be concluded as follows:

Table 8: Performance comparison with searched backbones on ADE20K validation dataset. Architectures were implemented with the same training recipe as [5]. All backbones were pretrained on ImageNet-1k, except for SETR, which was pretrained on ImageNet-21k dataset. $\star$ indicates the re-implementation results of important baseline methods with our recipe. Our results are highlighted in bold.

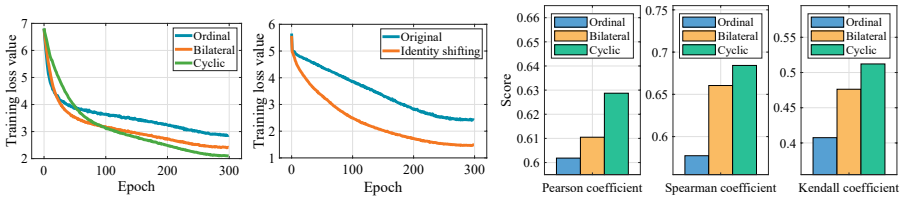
Backbone	Semantic FPN 80k [5]			Upernet 160k [25]		
	FLOPs (G)	Param (M)	mIoU (%)	FLOPs (G)	Param (M)	mIoU (%)
ResNet50 [17]	45	28.5	36.7	-	-	-
Twins-PCPVT-S [5]	40	28.4	44.3	234	54.6	46.2
Swin-T [25]	46	31.9	41.5	237	59.9	44.5
Twins-SVT-S* [5]	37	28.3	43.6	228	54.4	45.9
ViTAS-Twins-S	38	35.1	**46.6**	229	61.7	**47.9**
ResNet101 [17]	66	47.5	38.8	-	-	-
Twins-PCPVT-B [5]	55	48.1	44.9	250	74.3	47.1
Swin-S [25]	70	53.2	45.2	261	81.3	47.6
Twin-SVT-B* [5]	67	60.4	45.5	261	88.5	47.7
ViTAS-Twins-B	67	69.6	**49.5**	261	97.7	**50.2**
ResNetXt101 [50]	-	86.4	40.2	-	-	-
PVT-Large [5]	71	65.1	42.1	-	-	-
Twins-PCPVT-L [5]	71	65.3	46.4	269	91.5	48.6
Swin-B [25]	107	91.2	46.0	299	121	48.1
Twins-SVT-L* [5]	102	103.7	46.9	297	133	48.8
ViTAS-Twins-L	108	128.2	**50.4**	303	158.7	**51.3**
Backnone	PUP (SETR [66])			MLA (SETR [66])		
T-Large (SETR) [66]	-	310	50.1	-	308	48.6

Table 9: Ablation studies of the proposed ViTAS. We implemented the search on the ImageNet-1k set with 1.4G FLOPs budget. Weak_aug (WA), private token (PT), and Identity shifting (IF) in Twins space on ImageNet.

	None	PT	WA	IF	PT+WA	PT+IF	WA+IF	PT+WA+IF
Ordinal [60,4]	77.2	77.3	77.5	77.4	77.8	77.6	77.9	78.1
Bilateral [37]	77.6	77.7	78.0	77.8	78.1	77.9	78.2	78.5
Cyclic	77.9	78.2	78.7	78.6	78.8	78.4	79.0	79.4

- In the first few epochs (*e.g.*, ≤ 20), the ordinal superformer has the fastest convergence, then is the bilateral pattern, the last is our cyclic one.
- After a few epochs (*e.g.*, ≥ 100), the superformer with the cyclic pattern can be best trained with the lowest loss value, while the bilateral pattern has the second convergence speed, and ordinal pattern performs the worst for training superformer.

It is because the ordinal pattern has the largest bias in training channels. As shown in Figure 3a, a part of the channels converges the fastest in the first few epochs. The bilateral pattern performs similar due to no influence uniformity considered. However, after training more epochs, many channels do not obtain well treated in the superformer of the ordinal and bilateral patterns, thus they present larger average loss values than the cyclic one.



(a) Weight sharing. (b) Identity shifting. (c) Performance w.r.t different coefficients.

Fig. 3: Comparisons of superformer training loss values and performance comparison of coefficients. (a) Superformer training loss w.r.t. ordinal, bilateral, and cyclic weight sharing mechanisms on ImageNet-1k dataset. (b) Superformer training loss w.r.t. identity shifting and original setting on ImageNet-100. (c) Performance comparison of coefficients w.r.t. ordinal, bilateral, and cyclic mechanisms with 2000 sampled paths.

Effect of identity shifting strategy. As in Figure 3b, we present the training losses of the ViTAS using identity shifting strategy and original setting, respectively, with ImageNet-100 dataset [8,40]. With the redundancy paths removed, our method can converge to a much smaller loss than the original one, which indicates the proposed identity shifting strategy can promote the training of the superformer. Concretely, the training loss of original and identity shifting decrease to 2.4 and 1.5 at the final, respectively, which indicates that our method promote to better convergence for superformer. Moreover, the results trained from scratch of the searched architectures with identity shifting or original setting is 90.4% and 88.3%, respectively.

Performance comparison of AutoFormer [60,4], BCNet [37], and ViTAS w.r.t. weight sharing paradigm with $2,000$ sampled paths. To perform the search, we uniformly assign 8 budget range from 1G to 8G FLOPs, with 250 paths in each weight sharing mechanism. Generally, we assume the performance of architectures are positively correlated with FLOPs. Thus, we can obtain the scores of the three patterns w.r.t. Pearson, Spearman, and Kendall coefficients on different FLOPs groups. As shown in Figure 3c, our method achieves remarkable improvements comparing to the others, which indicates that our superformer can provide more precisely ranking for architectures. Details of coefficients are elaborated in Section A.5 of Appendix.

7 Conclusion

In this paper, we presented a vision transformer architecture search (*i.e.*, ViTAS) framework with the formulated cyclic weight sharing paradigm for the fair ranking of dimensions and also search efficiency. Besides, we propose the identity shifting strategy to arrange the the ID operation at the deeper layers for removing the redundant paths in the superformer. Moreover, we also investigated the training strategy of the superformer and proposed the weak augmentation strategy during search to boost the performance of ViTAS. Extensive experiments on ImageNet-1k, COCO2017, and ADE20k datasets w.r.t. Twins- and DeiT-based transformer space prove the effectiveness of our ViTAS in terms of performance and efficiency.

References

1. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)
2. Chen, C.F., Fan, Q., Panda, R.: Crossvit: Cross-attention multi-scale vision transformer for image classification. arXiv preprint arXiv:2103.14899 (2021)
3. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
4. Chen, M., Peng, H., Fu, J., Ling, H.: AutoFormer: Searching transformers for visual recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 12270–12280 (October 2021)
5. Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C.: Twins: Revisiting the design of spatial attention in vision transformers. arXiv preprint arXiv:2104.13840 (2021)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE transactions on evolutionary computation **6**(2), 182–197 (2002)
7. Deng, C., Yang, E., Liu, T., Tao, D.: Two-stream deep hashing with class-specific centers for supervised image search. IEEE transactions on neural networks and learning systems **31**(6), 2189–2201 (2019)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
10. Diamond, S., Boyd, S.: Cvxpy: A python-embedded modeling language for convex optimization. The Journal of Machine Learning Research **17**(1), 2909–2913 (2016)
11. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
12. Du, R., Xie, J., Ma, Z., Chang, D., Song, Y.Z., Guo, J.: Progressive learning of category-consistent multi-granularity features for fine-grained visual classification (2021)
13. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010)
14. Guo, Z., Zhang, X., Mu, H., Heng, W., Liu, Z., Wei, Y., Sun, J.: Single path one-shot neural architecture search with uniform sampling. In: European Conference on Computer Vision. pp. 544–560. Springer (2020)
15. Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., Wang, Y.: Transformer in transformer. arXiv preprint arXiv:2103.00112 (2021)
16. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
18. Huang, T., You, S., Wang, F., Qian, C., Zhang, C., Wang, X., Xu, C.: Greedynasv2: Greedier search with a greedy path filter. arXiv preprint arXiv:2111.12609 (2021)
19. Huang, T., You, S., Yang, Y., Tu, Z., Wang, F., Qian, C., Zhang, C.: Explicitly learning topology for differentiable neural architecture search. arXiv preprint arXiv:2011.09300 (2020)

20. Huang, T., You, S., Zhang, B., Du, Y., Wang, F., Qian, C., Xu, C.: Dyrep: Bootstrapping training with dynamic re-parameterization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 588–597 (2022)
21. Li, C., Tang, T., Wang, G., Peng, J., Wang, B., Liang, X., Chang, X.: BossNAS: Exploring hybrid cnn-transformers with block-wisely self-supervised neural architecture search (2021)
22. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
23. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
24. Liu, H., Simonyan, K., Yang, Y.: DARTS: differentiable architecture search. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9 (2019)
25. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14030 (2021)
26. Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., Zhang, C.: Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2736–2744 (2017)
27. Liu, Z., Sun, M., Zhou, T., Huang, G., Darrell, T.: Rethinking the value of network pruning. arXiv preprint arXiv:1810.05270 (2018)
28. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
29. Park, J., Boyd, S.: General heuristics for nonconvex quadratically constrained quadratic programming. arXiv preprint arXiv:1703.07870 (2017)
30. Paszke, A., Gross, S., Chintala, S., Chanan, G.: Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration. PyTorch: Tensors and dynamic neural networks in Python with strong GPU acceleration **6** (2017)
31. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: Improving language understanding with unsupervised learning. Tech. rep., OpenAI (2018)
32. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
33. Srinivas, A., Lin, T.Y., Parmar, N., Shlens, J., Abbeel, P., Vaswani, A.: Bottleneck transformers for visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16519–16529 (2021)
34. Su, X., Huang, T., Li, Y., You, S., Wang, F., Qian, C., Zhang, C., Xu, C.: Prioritized architecture sampling with monto-carlo tree search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10968–10977 (2021)
35. Su, X., You, S., Huang, T., Wang, F., Qian, C., Zhang, C., Xu, C.: Locally free weight sharing for network width search. arXiv preprint arXiv:2102.05258 (2021)
36. Su, X., You, S., Huang, T., Wang, F., Qian, C., Zhang, C., Xu, C.: Locally free weight sharing for network width search. arXiv preprint arXiv:2102.05258 (2021)
37. Su, X., You, S., Wang, F., Qian, C., Zhang, C., Xu, C.: Bcnet: Searching for network width with bilaterally coupled network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2175–2184 (2021)
38. Su, X., You, S., Zheng, M., Wang, F., Qian, C., Zhang, C., Xu, C.: K-shot nas: Learnable weight-sharing for nas with k-shot supernets. arXiv preprint arXiv:2106.06442 (2021)
39. Tang, Y., You, S., Xu, C., Han, J., Qian, C., Shi, B., Xu, C., Zhang, C.: Reborn filters: Pruning convolutional neural networks with limited data. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 5972–5980 (2020)
40. Tian, Y., Krishnan, D., Isola, P.: Contrastive multiview coding. In: ECCV. pp. 776–794 (2020)

41. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. arXiv preprint arXiv:2012.12877 (2020)
42. Wan, A., Dai, X., Zhang, P., He, Z., Tian, Y., Xie, S., Wu, B., Yu, M., Xu, T., Chen, K., et al.: Fbnetv2: Differentiable neural architecture search for spatial and channel dimensions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12965–12974 (2020)
43. Wang, C., Xu, C., Yao, X., Tao, D.: Evolutionary generative adversarial networks. IEEE Transactions on Evolutionary Computation **23**(6), 921–934 (2019). https://doi.org/10.1109/TEVC.2019.2895748
44. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. arXiv preprint arXiv:2102.12122 (2021)
45. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. arXiv preprint arXiv:2103.15808 (2021)
46. Xiao, T., Liu, Y., Zhou, B., Jiang, Y., Sun, J.: Unified perceptual parsing for scene understanding. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 418–434 (2018)
47. Xie, J., Ma, Z., Chang, D., Zhang, G., Guo, J.: GPCA: A probabilistic framework for Gaussian process embedded channel attention (2021)
48. Xie, J., Ma, Z., Lei, J., Zhang, G., Xue, J.H., Tan, Z.H., Guo, J.: Advanced dropout: A model-free methodology for Bayesian dropout optimization (2021)
49. Xie, J., Ma, Z., Xue, J.H., Zhang, G., Sun, J., Zheng, Y., Guo, J.: DS-UI: Dual-supervised mixture of Gaussian mixture models for uncertainty inference in image recognition **30**, 9208–9219 (2021)
50. Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1492–1500 (2017)
51. Xu, H., Su, X., Wang, D.: Cnn-based local vision transformer for covid-19 diagnosis. arXiv preprint arXiv:2207.02027 (2022)
52. Xu, H., Su, X., Wang, Y., Cai, H., Cui, K., Chen, X.: Automatic bridge crack detection using a convolutional neural network. Applied Sciences **9**(14), 2867 (2019)
53. Xu, H., Su, X., You, S., Huang, T., Wang, F., Qian, C., Zhang, C., Xu, C., Wang, D., Sowmya, A.: Data agnostic filter gating for efficient deep networks. In: ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3503–3507. IEEE (2022)
54. Xu, H., Wang, D., Sowmya, A.: Multi-scale alignment and spatial roi module for covid-19 diagnosis. arXiv preprint arXiv:2207.01345 (2022)
55. Yan, Z., Dai, X., Zhang, P., Tian, Y., Wu, B., Feiszli, M.: Fp-nas: Fast probabilistic neural architecture search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15139–15148 (2021)
56. Yang, Y., Li, H., You, S., Wang, F., Qian, C., Lin, Z.: Ista-nas: Efficient and consistent neural architecture search by sparse coding. Advances in Neural Information Processing Systems **33** (2020)
57. Yang, Y., You, S., Li, H., Wang, F., Qian, C., Lin, Z.: Towards improving the consistency, efficiency, and flexibility of differentiable neural architecture search. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6667–6676 (2021)
58. You, S., Huang, T., Yang, M., Wang, F., Qian, C., Zhang, C.: Greedynas: Towards fast one-shot nas with greedy supernet. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1999–2008 (2020)

59. You, S., Xu, C., Xu, C., Tao, D.: Learning from multiple teacher networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1285–1294 (2017)
60. Yu, J., Huang, T.: Autoslim: Towards one-shot architecture search for channel numbers. arXiv preprint arXiv:1903.11728 **8** (2019)
61. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z., Tay, F.E., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. arXiv preprint arXiv:2101.11986 (2021)
62. Zheng, M., Wang, F., You, S., Qian, C., Zhang, C., Wang, X., Xu, C.: Weakly supervised contrastive learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10042–10051 (2021)
63. Zheng, M., You, S., Huang, L., Wang, F., Qian, C., Xu, C.: Simmatch: Semi-supervised learning with similarity matching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14471–14481 (2022)
64. Zheng, M., You, S., Wang, F., Qian, C., Zhang, C., Wang, X., Xu, C.: Ressl: Relational self-supervised learning with weak augmentation. Advances in Neural Information Processing Systems **34**, 2543–2555 (2021)
65. Zheng, M., You, S., Wang, F., Qian, C., Zhang, C., Wang, X., Xu, C.: Relational self-supervised learning. arXiv preprint arXiv:2203.08717 (2022)
66. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6881–6890 (2021)
67. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 633–641 (2017)
68. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)