Vision Transformer Architecture Search

Xiu Su[1] **Shan You**[2,4*] **Jiyang Xie**[3] **Mingkai Zheng**[2] **Fei Wang**[2]
Chen Qian[2] **Changshui Zhang**[4] **Xiaogang Wang**[2,5] **Chang Xu**[1]
[1]School of Computer Science, Faculty of Engineering, The University of Sydney
[2]SenseTime Research
[3]Beijing University of Posts and Telecommunications
[4]Department of Automation, Tsinghua University,
Institute for Artificial Intelligence, Tsinghua University (THUAI),
Beijing National Research Center for Information Science and Technology (BNRist)
[5]The Chinese University of Hong Kong
xisu5992@uni.sydney.edu.au, xiejiyang2013@bupt.edu.cn
{youshan,zhengmingkai,wangfei,qianchen}@sensetime.com,
zcs@mail.tsinghua.edu.cn, xgwang@ee.cuhk.edu.hk,c.xu@sydney.edu.au,

Abstract

Recently, transformers have shown great superiority in solving computer vision tasks by modeling images as a sequence of manually-split patches with self-attention mechanism. However, current architectures of vision transformers (ViTs) are simply inherited from natural language processing (NLP) tasks and have not been sufficiently investigated and optimized. In this paper, we make a further step by examining the intrinsic structure of transformers for vision tasks and propose an architecture search method, dubbed ViTAS, to search for the optimal architecture with similar hardware budgets. Concretely, we design a new effective yet efficient weight sharing paradigm for ViTs, such that architectures with different token embedding, sequence size, number of heads, width, and depth can be derived from a single super-transformer. Moreover, to cater for the variance of distinct architectures, we introduce *private* class token and self-attention maps in the super-transformer. In addition, to adapt the searching for different budgets, we propose to search the sampling probability of identity operation. Experimental results show that our ViTAS attains excellent results compared to existing pure transformer architectures. For example, with 1.3G FLOPs budget, our searched architecture achieves 74.7% top-1 accuracy on ImageNet and is 2.5% superior than the currently baseline ViT architecture. Code is available at https://github.com/xiusu/ViTAS.

1 Introduction

Transformer, as a self-attention characterized neural network, has been widely leveraged for natural language processing tasks [5, 18, 19, 1]. Amazingly, recent breakthrough of vision transformer (ViT) [6, 27] further reveals the huge potential of transformers in computer vision tasks. With no use of inductive biases, self-attention layers in the transformer introduce a global receptive field, which conveys refresh solutions to process vision data. Following ViT, there have been quite a few works on vision transformers for a variety of tasks, such as image recognition [6, 27, 37, 8, 14, 32], object detection [38, 14], and semantic segmentation [14].

Despite the remarkable achievements of transformer in vision, the design of vision transformer architectures is still rarely investigated. Current vision transformers simply split an image into a sequence of patches (*i.e.*, tokens), and stack transformer blocks as natural language processing

*Corresponding author.

Preprint. Under review.

(NLP) tasks. Nevertheless, this vanilla protocol does not necessarily ensure the optimality for vision tasks. Stacking manner and intrinsic structure of transformer 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, and depth of the whole model. Therefore we raise questions that *What makes a better vision transformer? How can we obtain it?* With this aim, we propose to directly search for an optimal architecture for vision transformers by gathering all variable components as a search space. As a result, manual design effort will be saved and more promising architectures will be discovered.

Inspired by the success of one-shot neural architecture search (NAS) in convolutional neural networks (CNNs), we also leverage a weight sharing paradigm to efficiently implement the searching. In this way, different transformer architectures (fully) share their weights in a single one-shot network, dubbed *super-transformer*, and can be evaluated by inheriting the corresponding weights. However, transformers are no CNNs; we empirically find that transferring the one-shot weight sharing of CNNs into transformers just induces plain results, even worse than the baseline ViTs. Here we present a new weight-sharing paradigm customized for vision transformers. Instead of fully-sharing all the parametric operations, we privatize parts of them including class token and self-attention maps to cater for the variance of distinct architectures.

Based on the customized weight-sharing paradigm for super-transformer, we propose a Vision Transformer Architecture Search (ViTAS) method. ViTAS functions in a two-stage manner, *i.e.*, training the super-transformer by sampling architectures and then searching for the optimal one under certain hardware budgets via evolutionary algorithms. However, since the search space for ViTs is fairly huge, the stability of super-transformer training is horrible and often induces *performance collapse*. We argue that strong augmentation and regularization hurt the training of super-transformer, and totally random architecture sampling in super-transformer training affects optimal architecture search. Experimental results show that our ViTAS achieved new state-of-the-art performance under similar hardware budgets of FLOPs or GPU throughout. For example, our searched 1.3G FLOPs ViT architecture achieves 74.7% top-1 accuracy on ImageNet, which is 2.5% higher than that of the baseline ViT [27]. Besides, by examining the searched architectures, we make several observations which may help the design of ViTs. For example, it is better to assign a small output dimension for the first layer, but a large one for the latest.

Our main contributions can be summarized as follows.

- We define a search space for vision transformers, and design a new weight-sharing paradigm for one-shot super-transformer, which privatizes the weights of class token and self-attention maps for different patch sizes and heads.
- We propose a new vision transformer architecture search (ViTAS) method, and argue that leveraging weak augmentation and regularization matters to stabilize the training of super-transformer.
- Experimental results show our searched 1.3G FLOPs ViT architecture achieves 74.7% top-1 accuracy on ImageNet dataset, 2.5% higher than that of the baseline ViT. In addition, we also achieve 74.6% top-1 accuracy by searching a single block and repeatedly stacking it for constructing the ViT architecture.

2 Related Work

Vision Transformer. ViT was first proposed by Dosovitskiy et al. [6] to extend the applications of transformers into computer vision fields by cascading manually designed multilayer perceptrons (MLPs) and MHSA modules. Touvron et al. [27] introduced a teacher-student strategy and a distillation token into the ViT, namely data-efficient image transformers (DeiT). The DeiT evaluated on the same architecture with ViT under different strategies.

Meanwhile, other variants of ViT were proposed in recent months, which all introduced inductive bias and prior knowledge to extract local information for better feature extraction abilities. Tokens-to-Token (T2T) ViT [37] added a layer-wise T2T transformation and an efficient backbone with a deep-narrow structure in order to overcome limitations of local structure modeling in images and feature richness. Then, Han et al. [8] proposed to model both patch- and pixel-level representations by transformer-in-transformer (TNT). Swin Transformer [14] generated various patch scales by shifted windows for better representing highly changeable visual elements. Wu et al. [32] reintroduced

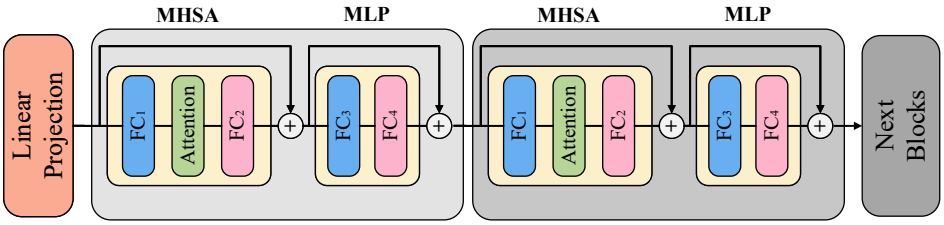


Figure 1: Illustration of the search space explanation of the ViT. An image is first fed into the linear projection (in orange) for patch and position embedding, and then inputted into cascaded blocks (in grey) for feature extraction. In each block, features are sequentially fed into an MHSA module and an MLP module with residual connections. Both of them contain two fully-connected (FC) layers (in blue and pink), while the former has an extra self-attention module (in green). Thus, patch size and output dimension of the linear projection, output dimension of the blue FC layers, and head number of the self-attention module can be searched. In the architecture-level, each block is searched independently, which they are same in the block-level. Furthermore, the depth of the ViT should be also selected in architecture-level.

Table 1: Macro search space for the ViTAS."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 Output Dim". A larger "Ratio" means a larger dimension.

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

convolutions into the ViT, namely convolutional vision transformer (CvT). All the aforementioned transformer structures were manually designed according to expert experience and grid search.

One-shot NAS Method. Differentiable architecture search (DARTS) [13, 33, 34, 10] first formulated the NAS task in a differentiable manner based on the continuous relaxation. Guo et al. [7] converted the search strategy into a single path one-shot (SPOS) framework with a uniform sampler to construct a simplified supernet. GreedyNAS [35] eased the burden of a supernet by encouraging it to focus more on evaluation of those potentially-good ones with a greedy search strategy while MCT-NAS [20] leveraged a Monte-Carlo tree to record the history of path quality. K-shot NAS [23] enhanced the evaluation ability by resorting to multiple supernets. Some work also attempted to investigate the channel dimension by direct searching [22, 21] or pruning from pretrained models [15, 26]. Moreover, FBNets [31, 30, 2] and EfficientNets [24, 25] focused more on jointly searching operations and channels for more efficient CNN design. However, the aforementioned one-shot NAS methods only focused on searching the optimal CNN architecture, but did not undertake in a transformer search space.

3 Architecture of Vision Transformer

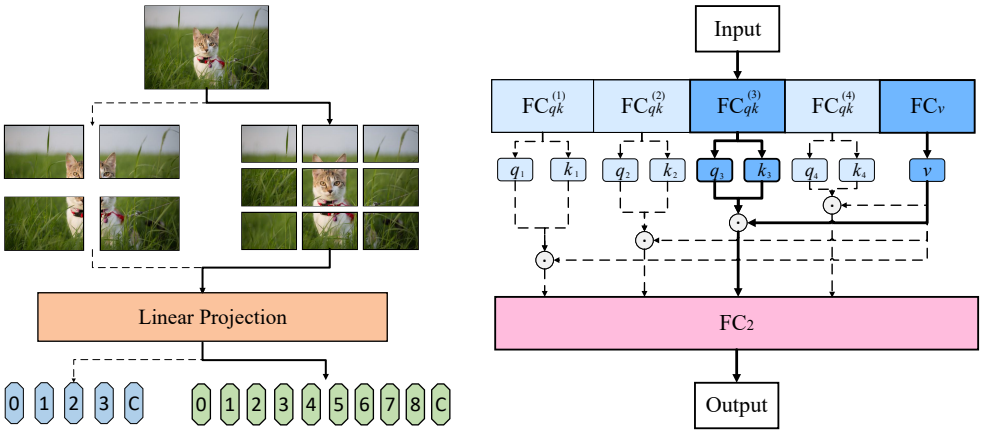
In this section, we elaborate the basic elements and the process of current vision transformer, then we designs the transformer search space for investigating promising architectures.

3.1 Statement of Vision Transformer

ViTs have obtained great performance and enjoyed efficient structures. Following the designs in NLP [5, 18, 19, 1], general ViTs [6, 27] manually chose the structures in block-level and stacked them to construct the whole models. Consider the input $X \in \mathbb{R}^{N \times D}$, where N and D are patch number and the dimension of a patch feature, respectively. In a standard block of [6], X is fed into an MHSA module and a two-layer MLP module with residual connections after each in a cascaded way, and LayerNorm (LN) is applied before each module. The two-layer MLP module contains a Gaussian error linear unit (GELU) as the nonlinear activation. We obtain the output $Y \in \mathbb{R}^{N \times D}$ by

$$X' = \text{MHSA}(\text{LN}(X)) + X, \tag{1}$$

$$Y = \text{MLP}(\text{LN}(X')) + X'. \tag{2}$$



(a) Private class tokens. (b) Private self-attention maps.

Figure 2: Illustration of the private class tokens and self-attention maps in the ViTAS. In the private class tokens, for two different patch sizes p (such as 4 and 9 in the figure), we assign two independent class tokens for each and obtain p patches and a private class token under one patch size setting. In the private self-attention maps, value v is shared among the four cases with the four head numbers, while q and k are obtained independently for all the cases.

In a general MHSA [6, 29], $Z = \mathrm{LN}(X)$ are linearly projected into three parts by an fully-connected (FC) layer $\mathrm{FC}_1(\cdot)$, each of which is of the same dimension as that of Z, i.e., the query $q \in \mathbb{R}^{N\times D}$, the key $k \in \mathbb{R}^{N\times D}$, and the value $v \in \mathbb{R}^{N\times D}$. Query and key together can measure the attentions of the features, and then be used to aggregate the value. q, k, and v are evenly separated into h groups for each head (h is the head number). After feeding each group into the corresponding self-attention (SA) in the i^{th} head, the output of the whole MHSA can be obtained by another FC layer $\mathrm{FC}_2(\cdot)$ to map the output feature dimension to $N \times D$. The whole MHSA can be formulated by

$$A_i = \mathrm{softmax}\left(q_i k_i^{\mathrm{T}}/\sqrt{D/h}\right), i = 1, \cdots, h, [q, k, v] = \mathrm{FC}_1(Z), \tag{3}$$

$$\mathrm{MHSA}(Z) = \mathrm{FC}_2\left([\mathrm{SA}_1(Z), \cdots, \mathrm{SA}_h(Z)]\right), \mathrm{SA}_i(Z) = A_i v_i, \tag{4}$$

where i is the head index. The overall architecture design of vision transformer is elegant and its effectiveness has already been well demonstrated in the literature [6, 27].

Most importantly, simply stacking the same block to build the vision transformer might not be the optimal solution. Different depths may require distinct capabilities to extract features of different levels (from low-level texture to high-level semantic ones). It is therefore to consider the design of the architecture from a global architecture-level (arch-level).

3.2 Towards Transformer Space

To explore the possibility of the optimal ViT architecture in the arch-level, we incorporate all the essential elements in our search space, including *head number*, *patch size*, *output dimension of each layer*, and *depth of the architectures*, as shown in Figure 1. The depth is only involved in the arch-level search space, and can be simply determined according to the pre-defined overall resource cost constraint in the block-level.

Table 1 describes the defined ViTAS search space in the arch-level. The block setting (the default order and numbers of MHSA and MLP) is inspired by [6, 27]. Meanwhile, the maximum depth is set as 16, and the patch size p is selected from the set $\{14, 16, 32\}$. The head number h in the MHSA is chosen within the set $\{3, 6, 12, 16\}$. The vanilla one-shot super-transformer search simply shares the class token and FC_1 in the MHSA for different architectures. The implementation way of sharing the class token is to pre-define the largest one and then dynamically select the required size of output [36, 30]. Meanwhile, the way of sharing FC_1 in the MHSA is to evenly divide its output features into h groups (h heads), while different h leads to different dimensions D/h of a group. Furthermore, in order to accommodate the output dimension, the dimension of each layer can be selected from a group of ten settings, *i.e.*, $\{\frac{i}{10}\}_{i=1}^{10}$. Given the defined search space, we encourage each block in the vision transformer to freely select their own optimal head numbers and output

Table 2: Ablation studies of the proposed ViTAS. We implemented the search on the ImageNet-100 (first 100 classes of the ImageNet) set and reduced the input resolution to the half (112×112). We report the top-1 retraining accuracy of the best architectures under 280M and 350M FLOPs budgets. "$\star$": original DeiT-Ti model (280M FLOPs) with half input resolution, we also uniformly scaled it to 350M FLOPs budget for fair comparison. DS: with dimension search only, ID: identity sampling probability search. "✓" indicates that we used the corresponding method. Best results are highlighted in bold.

#	Search	MHSA	Private Class Token	280M	350M
0$\star$				55.9%	56.2%
1	DS			54.0%	54.5%
2	NAS w/o ID			54.7%	55.0%
3	NAS w/o ID		✓	56.0%	56.6%
4	NAS		✓	56.6%	57.4%
5	NAS	q, k, v	✓	56.9%	58.5%
6	NAS	q, k	✓	56.2%	57.0%
7	NAS	v	✓	**57.6%**	**59.2%**

dimensions. As a result, the size of search space amounts to 1.3×10^{43} in the arch-level and 4×10^3 in the block-level.

4 Vision Transformer Architecture Search

Inspire by the success of CNN-based NAS, we first present the search process of vanilla one-shot NAS for the proposed transformer space. However, with this setting, one-shot NAS methods induce inferior searching performance. To accommodate the search of an optimal result, we propose a new weight-sharing paradigm customized for vision transformers. Besides, to boost the search of super-transformer, we investigate a stabilized training strategy for the super-transformer for fairly estimating the performance w.r.t. different architectures.

4.1 One-shot NAS for Transformer Space

The super-transformer in a huge search space cannot be fully trained due to resource limitation. It is difficult to optimize each setting in a whole super-transformer and/or retrain each of them to choose the best one. To construct a simplified super-transformer, a one-shot weight sharing NAS strategy can be considered in vision transformer architecture search (ViTAS). Define a transformer network $\mathcal{N}(\mathcal{A}, W)$ with search space $\mathcal{A}$ and model weights W of the super-transformer. We perform a two-stage optimization, which is formulated as

$$W_{\mathcal{A}}^* = \underset{W}{\operatorname{argmin}}\, loss_{\text{train}}\left(\mathcal{N}(\mathcal{A}, W_{\mathcal{A}})\right), \tag{5}$$

$$\alpha^* = \underset{\alpha \in \mathcal{A}}{\operatorname{argmax}}\, Acc_{\text{val}}\left(\mathcal{N}(\alpha, W_{\alpha}^*)\right), \tag{6}$$

where $loss_{\text{train}}$ is training loss, Acc_{val} is accuracy of validation set, $W_{\mathcal{A}}^*$ is a set of trained super-architecture weights, α is an architecture, and α^* is the searched optimal architecture. We first train the super-transformer by uniformly sampling different α from $\mathcal{A}$, and then search the optimal architecture α^* according to W^*. After these, the selected α^* will be retrained for evaluation.

Super-transformer training. Following the one-shot framework, we sample one architecture from the super-transformer in each iteration and train it with a batch of data. For fairly training all the architectures, we uniformly sample each of them from the search space.

Optimal Architecture Searching. We conduct the optimal architecture searching process by evolutionary algorithm. Here, we leverage the multi-objective NSGA-II algorithm [3] for the searching. We set the population size as 50 and the generation number as 40. With different budgets, *i.e.*, FLOPs and throughput, we randomly select a group of 50 budget-satisfied architectures as the first generation and find top-20 architectures on validation accuracy as parents to generate the next generation by mutation and crossover. After 40 rounds, we choose the optimal architecture with highest validation accuracy.

4.2 Weight-sharing with Privatization

We propose the following two privatization techniques to address two fatal issues, including patch information and positional information collapse, caused by directly introducing the naive one-shot weight sharing NAS into ViTAS.

Private class tokens. The vanilla ViTs perform various tasks with a parametric class token for a single network training. The class token is concatenated and optimized simultaneously with the input feature maps, then will be leveraged for the output calculation. Although it perfectly works for a single architecture, the ViTAS performs on a collection of the entire search space, where transformer architectures in it with different patch sizes may account to different semantic information. It may be damaged during patch size search, as different patch sizes determine the distinct granularities of discriminative features in each local part.

As illustrated in Figure 2a, we set the ViT architectures inherit one class token of its own patch size. This is because a class token is a learnable parameter vector in the ViTAS, which is optimized with the input features and exchanges token level information in the MHSA. Even with a little change to the patch size p, the number of patch tokens N varies greatly, *i.e.*, $N = 196$ for $p = 16$ and $N = 256$ for $p = 14$. Therefore, to promote the search of the ViTAS, we allow each patch size to have its private class token. Second, to promote the optimization of the parametric one, each class token should be updated with sufficient times. Hence, we limit each patch size to have only one class token.

Table 2 demonstrates that adding the private class tokens can remarked increase the top-1 retraining accuracy from 55.0% (group 2) to 56.6% (group 3). Thus, the private class tokens for each patch size can overcome the drawbacks.

Private self-attention for multiple heads. The vanilla ViTs conduct an MHSA module in a transformer block to adapt the global mutual relations of patch tokens. The ViTAS involves the architectures of the total search space and tends to fully share q, k, and v for different head numbers, with each one inheriting a part of the weights. Generally, q and k jointly determine the positional relations among the head number, and then access the representation features through their corresponding v. For the architectures with different head number h, the sequential dimension of q and k is evenly partitioned to D/h, which indicates different location information w.r.t. the head number. Thus, sharing them will affect the localization under different head numbers.

We allow each head to privatize its q and k for the auto-correlation positioning, as shown in Figure 2b. On the other heads, v presents the representation of the feature maps, which indicates that they should share similar values w.r.t. head number. In the ViTAS, $\text{FC}_1(\cdot)$ in (3) can be divided into $\text{FC}_{qk}(\cdot)$ and $\text{FC}_v(\cdot)$. The MHSA with $h^{(j)}, j = 1, \cdots, 4$ is formulated by

$$A_i^{(j)} = \text{softmax}\left(\frac{q_i^{(j)}(k_i^{(j)})^{\text{T}}}{\sqrt{D/h^{(j)}}}\right), i = 1, \cdots, h^{(j)}, [q^{(j)}, k^{(j)}] = \text{FC}_{qk}^{(j)}(Z), \tag{7}$$

$$\text{MHSA}^{(j)}(Z) = \text{FC}_2\left([\text{SA}_1^{(j)}(Z), \cdots, \text{SA}_h^{(j)}(Z)]\right), \text{SA}_i^{(j)}(Z) = A_i^{(j)} v_i, v = \text{FC}_v(Z). \tag{8}$$

As shown in Table 2, sharing v only (group 7) can achieve higher top-1 retraining accuracy than those of sharing q, k (group 6) or all of them (group 5), respectively, with 2.2% or 0.7% improvements. It can also obtain better performance than that of the group 4 with independent q, k, v for each head number (1.8% higher). Sharing v only is natural sense for the self-attention, as we consider fix the gallery (*i.e.*, v) and find different keys and queries under distinct head numbers for various numbers of local patterns.

4.3 Stabilizing the Training of Super-transformer

Training the super-transformer is for fairly estimating each architecture's performance, which is essential for the next optimal architecture searching stage. Therefore, we require an efficient and easy strategy for steady training, rather than a tricky one. The complex training recipe of DeiT is unsuitable for super-transformer training (see group 0 in Table 3). Furthermore, we argue that pre-set budgets should be used for constraining the sampling distribution for α, rather than a condition in the optimal architecture searching, since the budgets of the architectures in the search space vary in a huge range. Thus, we consider three super-transformer training strategies for preventing it from unsteady, including weak augmentation & regularization and identity probability search.

Table 3: Ablation studies of training recipe of the ViTAS. "✓" indicates that we used the corresponding method. We implemented the search on the ImageNet100 set and reduced the input resolutions to the half (112×112). We report the top-1 accuracy of the best architectures in both ViTAS (*i.e.*, searching) and retraining. "$\star$" indicates that the ViTAS is collapsed with the corresponding recipe. Best results are highlighted in bold.

	Data augmentation				Regularization			ViTAS	Retraining
#	Rand-Augment	Mixup	CutMix	Color Jitter	Erasing	Stoch Depth	Repeated Aug	Top-1	Top-1
$0^\star$	✓	✓	✓	✓	✓	✓	✓	4.2%	13.6%
$1^\star$	✓	✓	✓	✓	✓		✓	4.7%	14.5%
$2^\star$	✓	✓	✓	✓	✓			6.1%	16.3%
3	✓			✓	✓			7.5%	55.8%
4				✓	✓			11.6%	56.7%
5		✓			✓			12.3%	57.1%
6			✓		✓			11.4%	**57.6%**
7					✓			12.7%	57.3%
8								**13.6%**	56.8%

Table 4: 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	WD	Warmup	LR Scheduler	Cutmix	Erasing
300	1024	AdamW	0.001	cosine	0.05	5	0.1	1.0	0.25

Weak augmentation & regularization. We explored the super-transformer training strategy of the ViTAS, including data augmentation and regularization. With the same search space as in Section 3.1, we conducted the evaluations with ImageNet100 set and half input resolutions to accelerate the fast validation, and report the retraining accuracy of the searched top-1 architecture. Compare to a single network, the super-transformer training is much harder to converge than a vanilla ViT, which needs a simply yet effective training strategy.

From Table 3, as group 0, the super-transformer is broken down with the default DeiT training strategy [27]. To facility the search, we first removed the stochastic depth, since our proposed identity probability search performs the similar effect in the ViTAS training. Then, we gradually dropped other data augmentation and/or regularization and find that only CutMix and Erasing can promote retraining accuracy with searching a better architecture, although the search accuracy under the super-transformer decreased slightly. In addition, removing all the data augmentation and regularization can partly improve the search accuracy, but the retraining one reduced.

Table 4 presents the ViTAS training recipe for our experiments. For the searched top-1 ViT architectures, we followed the same retraining strategy as DeiT [27] to retrain on the ImageNet. Notably, for the architectures smaller than 1.3G FLOPs, we removed CutMix during retraining, since the capacity of these models is too weak to withstand excessive augmentations.

Constrained sampling for budget alignment. With the pre-set search space, the sampled budgets of the architectures vary in a huge range, *i.e.*, 7.9M and 6.6G FLOPs for the smallest and the largest ones, respectively. In this way, the average sampled budgets of the ViTAS have a large gap from the target budgets, which may lead to the super-transformer trapped in a suboptimal solution. Fortunately, we empirically find that the average sampled budgets of the ViTAS is directly correlated with the sampling probability of the identity operations, which is a non-parametric operation and will not involve the unfair training issues of the ViTAS. Ideally, with the pre-set budgets B, we aim to find the empirical *identity sampling probability search* by $B = \mathbb{E}_{\alpha \in S(p)} [\![\mathbb{B}(\alpha)]\!]$. The role of the identity probability search is to align the expected budges of the selected architectures to the pre-set budgets, which prevents the super-transformer training from misleading convergence. With the proposed identity probability search, the top-1 retraining accuracy of searched architecture is been boosted by 0.8% (from group 3 to 4 in Table 2).

5 Experimental Results

We perform the architecture search with the challenging ImageNet dataset [4]. 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. To compare with existing pure ViT methods [6], we report the accuracy on

Table 5: Searched ViT architectures w.r.t. different FLOPs and GPU throughput on ImageNet. The search targets are only constrained with FLOPs except of ViTAS-E. "⋆": architectures that involves inductive bias. Best results are highlighted in bold. "†": architectures that are searched with GPU throughput.

Method	FLOPs (G)	throughput (image/s)	Params (M)	Top-1 (%)	Top-5 (%)
ResNet-18⋆	1.8	4458.4	12	69.8	89.1
ResNet-50⋆	4.1	1226.1	25	76.2	91.4
DeiT-Ti	1.3	2728.5	5	72.2	91.3
ViTAS-A	0.9	4884.1	3.7	71.2	89.9
ViTAS-B	1.0	5349.9	4.3	71.7	90.1
ViTAS-C	1.3	2646.3	5.6	74.7	91.6
ViTAS-D	1.6	2031.2	7.0	76.2	92.5
ViTAS-E†	2.7	1742.2	12.6	**77.4**	**93.8**

Table 6: Transferability of the ViTAS to downstream tasks. Top-1 accuracies for each dataset are reported. FLOPs and throughput are also compared. "⋆": architectures that involves inductive bias. Best results are highlighted in bold.

Methods	FLOPs (G)	throughput (image/s)	ImageNet	CIFAR-10	CIFAR-100	Aircraft	Cars
ResNet-18⋆	1.8	4458.4	69.8	95.0	78.0	69.2	77.0
ResNet-50⋆	4.1	1226.1	76.2	96.9	83.3	**79.1**	**86.9**
DeiT-Ti	1.3	2728.5	72.2	97.4	85.6	72.2	83.7
ViTAS-A	0.9	4884.1	71.2	97.0	83.7	67.1	79.2
ViTAS-C	1.3	2646.3	74.7	97.6	85.8	72.8	84.1
ViTAS-D	1.6	2008.2	**76.2**	**97.9**	**86.6**	73.8	85.6

the original validation dataset. All experiments are implemented with PyTorch [17] and trained on NVIDIA Tesla V100 GPUs.

5.1 Efficient search of ViTAS on ImageNet

In Table 5, we compare our results with recent precedent ViT [6] and DeiT [27] (same model structure to the ViT with better training strategy). To fully illustrate the current progress of the ViT, we also include landmark CNN models as the baselines [9]. Notably, the ViT generally works well in the large-scale models. However, due to the less of inductive biases, the pure ViT fails to compare with the CNN-based methods [6, 27]. In this way, we aim to explore the possibility of searching good architectures with small budgets. To evaluate our methods with other existing algorithms, we present the search results with two kinds of budgets, *i.e.*, FLOPs and GPU throughput.

With 1.3G FLOPs and similar GPU throughput, our searched ViTAS-C achieves 2.5% superior on Top-1 accuracy than DeiT-Ti, which indicates the effectiveness of our proposed ViTAS method. Besides, with only 2.7G FLOPs budget, our ViTAS-D surpasses ResNet-50 (4.1G) by 1.2%, which indicates that our searched ViT architectures can achieve better results than ResNet in terms of efficiency and performance.

5.2 Transferability of ViTAS to Downstream Tasks

Besides for searching the optimal ViT architectures on the ImageNet set, we also verified the generalization ability of the ViTAS by transferring the searched architectures to other datasets. We evaluated this transferability by fine-tuning our results on downstream datasets [12, 11, 16]. Table 6 compares the transfer results of the ViTAS with the precedent DeiT and CNN-based ResNet-18 for presenting our improvements. Generally, we pretrained all models on the ImageNet set with 300 epochs, and then fine-tuned 150 epochs on the downstream tasks. From Table 6, with aligned 1.3G FLOPs budget, our searched ViTAS-C can perform better transfer results than that of DeiT-Ti, which verifies the transferability of our searched models with ViTAS.

5.3 Searching in Block-level

Although the inspiring results from the ViTAS, the searched ViT archiectures are complex and hard to remember for researchers. For the practicality of the ViTAS, we wonder *whether we can search*

Table 7: Searched ViT architectures w.r.t.different FLOPs and GPU throughput on the ImageNet dataset. The search targets were only constrained with FLOPs budgets. Our models were constructed by stacking the two searched blocks, respectively. "⋆": architectures that involves inductive bias. Best results are highlighted in bold.

Methods	FLOPs (G)	throughput (image/s)	Params (M)	Top-1 (%)	Top-5 (%)
ResNet-18⋆	1.8	4458.4	12	69.8	89.1
DeiT-Ti	1.3	2728.5	5	72.2	91.3
ViTAS-Block	1.3	3247.5	6.0	**74.6**	**92.0**

Table 8: Structure of the searched block in the block-level with the ViTAS under 1.3G FLOPs budget. In retraining, we simply stacked the searched block to construct the whole ViT architecture.

Number	Type	Patch size / #Heads	Output Dim
1	Embedding	16	230
12	MHSA	3	432
	MLP	-	720

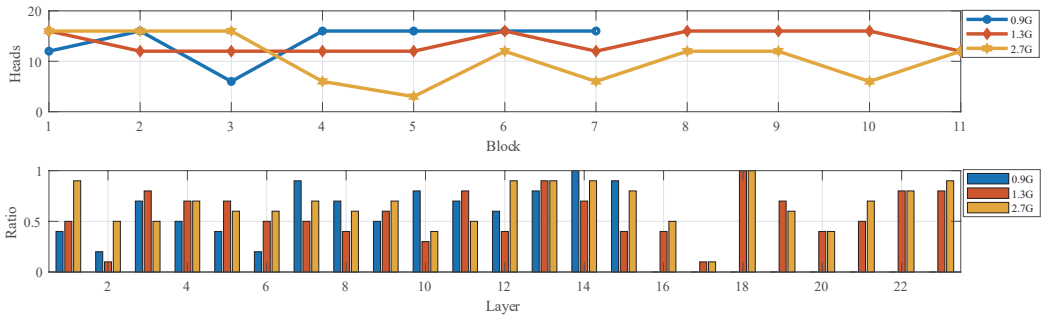


Figure 3: Visualizations of three searched architectures (ViTAS-A (0.9G FLOPs, in blue), ViTAS-C (1.3G FLOPs, in red), and ViTAS-E (2.7G FLOPs, in yellow), including head numbers of each block and ratios of output dimensions of FC layers. We should note that "Ratio" of layer 1 is for the FC layer in the linear projection and the i^{th} block contains "Ratio" of layer $2i$ and $2i+1$. The identity operations are ignored and we only show the parametric blocks and layers for better illustration.

for one block in the block-level that can achieve decent performance by simply stacking it. With this goal, we restricted all transformer blocks in a single architecture (*i.e.*, a cell) to have the same structure, including head number and output dimension, with steady patch size as 16. Therefore, the space is shrunk to 4000 architectures, *i.e.*, 10 cases of embedding layer output dimension, 4 optional head number, 10 cases for MHSA input dimension, and 10 cases for multi layer perception output dimension. With these settings, it is very tough to find the block, since the too small search space limits the upper bound of the ViT architectures and we may need to find almost the best one for boosting performance from the ViT.

With 1.3G FLOPs budget, we are surprised to find that DeiT-tiny is exactly the same architecture to the rank-3 one searched in the ViTAS, which proves that pioneer architectures have already been carefully designed in [6]. From Table 8, We present the rank-1 one of the ViTAS. With the same strategies as in Section 5.1 and 5.2, we show the retraining results of stacking the block in Table 7. As a result, we achieve 2.4% on top-1 accuracy superior than the DeiT.

5.4 Visualization and Interpretation of Searched ViT Architectures

In this section, we discuss the searched ViT architectures. For intuitively understanding, we visualize our searched three ViT architectures with various FLOPs in Figure 3 as examples. From Figure 3, we summarize the following four experiential suggestions for further ViT design.

- For a tiny architecture with a small budget, less blocks is more suitable, *i.e.*, it need a shallow but wide structure, but not a thin and deep one. For example, compared with the 11-block ViTAS-C (1.3G FLOPs), the ViTAS-A (0.9G FLOPs) consists of only 7 blocks.
- We know that head number does not affect FLOPs. In this case, we can find that blocks tend to select 12 or 16 heads in their MHSA modules, when we searched with FLOPs

budgets. This proves that a larger head number can actually improve the performance of a ViT, although this will decrease the speed (GPU throughput).

- For the first block of the ViT, the first FC layer tends to retain a smaller output dimension (see "Ratio" of layer 2) than the latter layers. This is because the output features of the linear projection are not robust enough and the model does not need too much dimension for information processing.
- The last blocks in each architecture maintain almost all the dimensions to suit the tasks' requirements. We know that more features may lead to better classification accuracy, which is similar to that in ResNet.

6 Conclusion

In this paper, we defined a search space for vision transformers (ViTs), and designed a new weight-sharing paradigm for one-shot super-transformer, which privatizes the weights of class token and self-attention maps for different patch sizes and heads. For helping the design of ViTs, we proposed a new vision transformer architecture search (ViTAS) method, and argued that leveraging weak augmentation and regularization matters to stabilize the training of super-transformer. Experimental results proves the effectiveness of our ViTAS in terms of performance and efficiency.

References

[1] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1, 3

[2] Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Bichen Wu, Zijian He, Zhen Wei, Kan Chen, Yuandong Tian, Matthew Yu, Peter Vajda, et al. Fbnetv3: Joint architecture-recipe search using neural acquisition function. *arXiv preprint arXiv:2006.02049*, 2020. 3

[3] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002. 5, 12

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 7

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 3

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 2, 3, 4, 7, 8, 9, 13

[7] 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 *European Conference on Computer Vision*, pages 544–560. Springer, 2020. 3

[8] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *arXiv preprint arXiv:2103.00112*, 2021. 1, 2

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 8

[10] Tao Huang, Shan You, Yibo Yang, Zhuozhuo Tu, Fei Wang, Chen Qian, and Changshui Zhang. Explicitly learning topology for differentiable neural architecture search. *arXiv preprint arXiv:2011.09300*, 2020. 3

[11] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV workshops*, 2013. 8

[12] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55, 2014. 8

[13] 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*. 3

[14] 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. *arXiv preprint arXiv:2103.14030*, 2021. 1, 2

[15] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017. 3

[16] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 8

[17] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 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. 8

[18] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. 2018. 1, 3

[19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1, 3

[20] Xiu Su, Tao Huang, Yanxi Li, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. Prioritized architecture sampling with monto-carlo tree search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10968–10977, 2021. 3

[21] Xiu Su, Shan You, Tao Huang, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. Locally free weight sharing for network width search. *arXiv preprint arXiv:2102.05258*, 2021. 3

[22] Xiu Su, Shan You, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. Bcnet: Searching for network width with bilaterally coupled network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2175–2184, 2021. 3

[23] Xiu Su, Shan You, Mingkai Zheng, Fei Wang, Chen Qian, Changshui Zhang, and Chang Xu. K-shot nas: Learnable weight-sharing for nas with k-shot supernets. *arXiv preprint arXiv:2106.06442*, 2021. 3

[24] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019. 3

[25] Mingxing Tan and Quoc V Le. Efficientnetv2: Smaller models and faster training. *arXiv preprint arXiv:2104.00298*, 2021. 3

[26] Yehui Tang, Shan You, Chang Xu, Jin Han, Chen Qian, Boxin Shi, Chao Xu, and Changshui Zhang. Reborn filters: Pruning convolutional neural networks with limited data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5972–5980, 2020. 3

[27] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020. 1, 2, 3, 4, 7, 8, 13

[28] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021. 13

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 4

[30] Alvin Wan, Xiaoliang Dai, Peizhao Zhang, Zijian He, Yuandong Tian, Saining Xie, Bichen Wu, Matthew Yu, Tao Xu, Kan Chen, 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*, pages 12965–12974, 2020. 3, 4

[31] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10734–10742, 2019. 3

[32] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021. 1, 2

[33] Yibo Yang, Hongyang Li, Shan You, Fei Wang, Chen Qian, and Zhouchen Lin. Ista-nas: Efficient and consistent neural architecture search by sparse coding. *Advances in Neural Information Processing Systems*, 33, 2020. 3

[34] Yibo Yang, Shan You, Hongyang Li, Fei Wang, Chen Qian, and Zhouchen Lin. 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*, pages 6667–6676, 2021. 3

[35] Shan You, Tao Huang, Mingmin Yang, Fei Wang, Chen Qian, and Changshui Zhang. Greedynas: Towards fast one-shot nas with greedy supernet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1999–2008, 2020. 3

[36] Jiahui Yu and Thomas Huang. Autoslim: Towards one-shot architecture search for channel numbers. *arXiv preprint arXiv:1903.11728*, 8, 2019. 4

[37] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021. 1, 2

[38] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 1

A Details of Evolutionary Search

To accommodate the huge search space (*i.e.*, 1.3×10^{43}) in the arch-level and boost the search efficiency, we leverage the multi-objective NSGA-II [3] to implement the evolutionary search, which is easy to incorporate FLOPs and GPU throughput constraints. In detail, we set the population size as 50 and generation number as 40, which amounts to 2000 architectures in the ViTAS. To perform the search, we first random select 50 architectures within the budgets as the initial population. Then in each iteration, we select the top-20 architectures in performance as parents to generate architectures for next generation via mutation and crossover. We infer each architecture with the super-transformer and the local validation set. After search, we only retrain the architecture with the highest local validation accuracy from scratch and report its performance.

B More Detailed Results on ImageNet for Table 5

To investigate the effect of ViTAS, we further implemented the search under 900 GPU throughput budget with the same strategies reported in our paper. Since 900 GPU throughput is a relatively large hardware statistics for those architectures in the current search space in Table 1, we proposed to customize its corresponding search space with doubling the maximum dimensions for each block, as shown in Table 9.

Table 9: Macro search space for the ViTAS. "TBS" indicates that the layer is searched from vanilla ViT block or identity operation for depth search. "Ratio" means the reduction ratio from the "Max Output Dim". A larger "Ratio" means a larger dimension. The changed part from Table 1 is highlighted in bold.

Number	Search OP	Type	Patch size / #Heads	Max Output Dim	Ratio
1	False	Linear Projection	$\{14, 16, 32\}$	**768**	$\{i/10\}_{i=1}^{10}$
16	TBS	MHSA	$\{3, 6, 12, 16\}$	**2880**	$\{i/10\}_{i=1}^{10}$
		MLP	-	**2880**	$\{i/10\}_{i=1}^{10}$

In detail, we present the searched results in Table 10. With similar GPU throughput to the DeiT-S, our searched ViTAS-F achieves 80.5% on Top-1 accuracy, which surpasses the DeiT-S by 0.7%.

C Mean and deviations of performance of searched architectures

We also report means and deviations of accuracies of the searched ViT architectures on the ImageNet in Table 11. We repeatedly conducted experiments for three times with different random seeds.

Table 10: Searched ViT architectures w.r.t. different FLOPs and GPU throughput on the ImageNet. The search targets are only constrained with FLOPs except of ViTAS-E and ViTAS-F. "⋆": architectures that involve inductive bias. Best results are highlighted in bold. "†": architectures that are searched with GPU throughput.

Method	FLOPs (G)	throughput (image/s)	Params (M)	Top-1 (%)	Top-5 (%)
ResNet-18⋆	1.8	4458.4	12	69.8	89.1
ResNet-50⋆	4.1	1226.1	25	76.2	91.4
DeiT-Ti	1.3	2728.5	5	72.2	91.3
DeiT-S	4.6	940.4	22	79.8	95.0
ViTAS-A	0.9	4884.1	3.7	71.2	89.9
ViTAS-B	1.0	5349.9	4.3	71.7	90.1
ViTAS-C	1.3	2646.3	5.6	74.7	91.6
ViTAS-D	1.6	2031.2	7.0	76.2	92.5
ViTAS-E†	2.7	1742.2	12.6	77.4	93.8
ViTAS-F†	6.0	909.4	27.6	**80.5**	**95.1**

Table 11: Means and standard deviations of accuracies of the searched ViT architectures on the ImageNet.

Method	Top-1 (%)	Top-5 (%)
ViTAS-A	71.2 ± 0.12	90.0 ± 0.11
ViTAS-C	74.6 ± 0.08	91.6 ± 0.12
ViTAS-E	77.4 ± 0.10	93.7 ± 0.09
ViTAS-F	80.5 ± 0.06	95.1 ± 0.07

D Transferability of ViTAS to CaiT

We notice that CaiT [28] proposed a new method for the pure ViT architectures and can achieve good performance. We further examine the transferred results of our searched ViTAS-Block with CaiT, including class-attention layers and learnable layer scale parameters. As illustrated in Table 12, with the same FLOPs budget, our searched ViTAS-Block surpasses the vanilla DeiT-Ti by 1.9% on Top-1 accurcy.

Table 12: Transferability of ViTAS to CaiT. † indicates that the model follow CaiT method.

Method	FLOPs (G)	Params (M)	Top-1 (%)	Top-5 (%)
CaiT (DeiT-Ti)†	1.3	5.0	72.6	90.9
CaiT (ViTAS-Block)†	1.3	6.0	**74.5**	**92.0**

E Searching in Block level with Same Budgets of DeiT-S

Searching a simple yet efficient ViT architectures in the block-level can promote the practical use of the ViTAS. To further examine the search of block level architectures, we implemented the search with a similar budget to the DeiT-S with the search space in Table 9. As shown in Table 13, we surprisingly find that our searched block-level Top-1 architecture has almost the same structure with that of the DeiT-S. We believe that this architecture should be carefully designed in [6, 27].

F Intuition of private (q, k) of MHSA in ViTAS

In this section, we examine the effect of (q, k) of MHSA w.r.t. different heads number. To achieve this aim, with DeiT-S architecture, we present the attention map A of the product of (q, k) w.r.t. 6 or 16 heads number. As shown in Figure 4, it is very different between the values of the attention maps of 6 and 16 heads, which means that the self-attention modules with different head numbers focus on different local parts. Therefore, we should not share (q, k), but privatize them instead.

Table 13: Structure of the searched block in the block-level with the ViTAS under 5.0G FLOPs budget. In retraining, we simply stacked the searched block to construct the whole ViT architecture.

Number	Type	Patch size / #Heads	Output Dim
1	Embedding	16	384
12	MHSA	6	1152
	MLP	-	1728

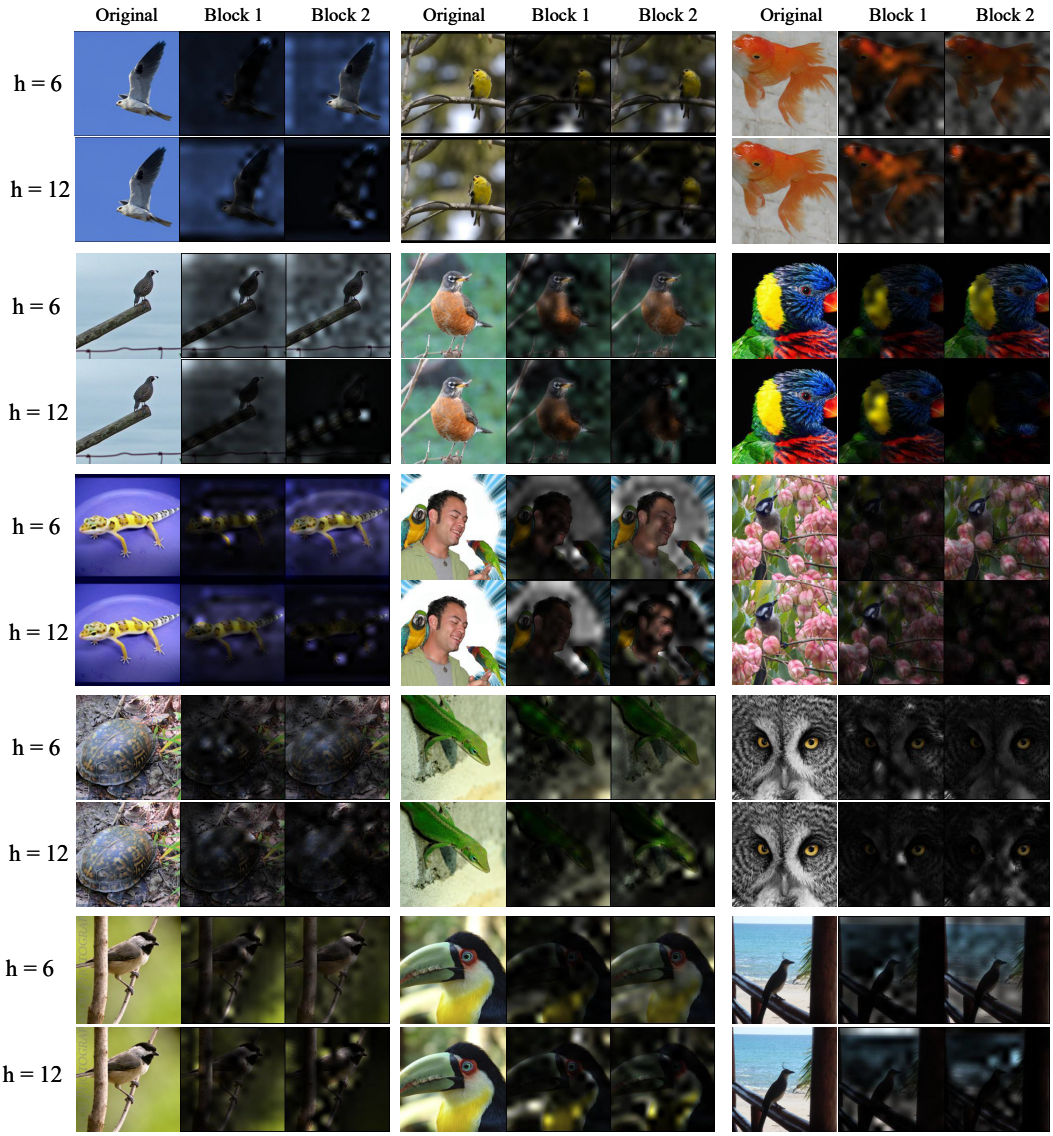


Figure 4: Visualizations of (q, k) of MHSA with different head numbers. Here, we select the first two blocks as an example. For the corresponding blocks, with head number h as 6 or 16, we illustrate average attention map A in (7) of different heads. In a group, the three columns indicate the original images, and A of the first and the second blocks, respectively, while the two rows indicate results of 6 or 16 heads, respectively. Brightness of the visualizations means values of the pixels in A.