Bridging the Gap Between Vision Transformers and Convolutional Neural Networks on Small Datasets

Anonymous Author(s)
Affiliation
Address
email

Abstract

There still remains extreme performance gap between Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs) when training from scratch on small datasets, which is concluded to the lack of inductive bias. In this paper, we further consider this problem and point out two weakness of ViTs in inductive biases, that is, the **spatial relevance** and **diverse channel representation**. First, on spatial aspect, objects are locally compact and relevant, thus fine-grained feature needs to be extracted from a token and its neighbours. While the lack of data hinders ViTs to attend the spatial relevance. Second, on channel aspect, representation exhibits diversity on different channels. But the scarce data can not enable ViTs to learn strong enough representation for accurate recognition. To this end, we propose Dynamic Hybrid Vision Transformer (DHVT) as the solution to enhance the two inductive biases. On spatial aspect, we adopt a hybrid structure, in which convolution is integrated into patch embedding and multi-layer perceptron module, forcing the model to capture the token features as well as theirs neighbouring features. On channel aspect, we introduce a dynamic feature aggregation module in MLP and a brand new "head token" design in multi-head self-attention module to help re-calibrate channel representation and make different channel group representation interacts with each other. The fusion of weak channel representation forms a strong enough representation for classification. With this design, we successfully eliminate the performance gap between CNNs and ViTs, and our DHVT achieves a series of state-of-the-art performance with a lightweight model, 85.68% on CIFAR-100 with 22.8M parameters, 82.3% on ImageNet-1K with 24.0M parameters. Code will be released if accepted.

1 Introduction

After a long-term domination by Convolutional Neural Networks (CNNs) in Computer Vision (CV) field, these years have witnessed the rapid growth of another promising alternative architecture paradigm, Vision Transformers (ViTs). They have already exhibited great performance in many vision tasks, such as image classification [1, 2, 3, 4, 5], object detection [6, 7, 8], segmentation [9, 10] and image generation [11, 12].

ViT [1] is the pioneering model that brings Transformer architecture [13] from Natural Language Processing (NLP) into CV. It has higher performance upper bound than standard CNNs, while it is at the cost of expensive computation and extremely huge amount of training data. The vanilla ViT needs to be firstly pre-trained on the huge dataset JFT-300M [1] and then fine-tuned on the common dataset ImageNet-1K [14]. Under this experimental setting, it shows higher performance than standard CNNs. However, when training from scratch on ImageNet-1K only, the accuracy is much lower. From the practical perspective, most of the datasets are even smaller than ImageNet-1K, and not all the researchers can hold the burden of pre-training their own model on large datasets and

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

then fine-tune on the target small datasets. Thus, an effective architecture for training ViTs from scratch on small datasets are demanded.

Recent works [15, 16, 17] explore the reasons for the difference in data-efficiency between ViT and CNNs, and draw a conclusion to the lack of inductive bias. In [15], it points out that *with not enough data, ViT does not learn to attend locally in earlier layers.* And in [16], it says that *the stronger the inductive biases, the stronger the representations. Large datasets tend to help ViT learn strong representations. Locality constraints improve the performance of ViT.* Meanwhile in recent work [17], it demonstrates that *convolutional constraints can enable strongly sample-efficient training in the small-data regime.* The insufficient training data makes ViT hard to derive the inductive bias of attending locality, thus many recent works strive to introduce local inductive bias by integrating convolution into ViTs [5, 2, 18, 19, 20] and modify it to hierarchical structure [21, 22, 3, 4, 23], making ViTs more like traditional CNNs. This style of hybrid structure shows comparable performance with strong CNNs when training from scratch on medium dataset ImageNet-1K only. But the performance gap on much smaller datasets still remains.

Here, we consider that the scarce training data weakens the inductive biases in ViTs. Two kinds of inductive bias need to be enhanced and better exploited to improve the data-efficiency, that is, the **spatial relevance** and **diverse channel representation**. **On spatial aspect**, tokens are relevant and object are locally compact. Important fine-grained low-level feature needs to be extracted from the token and its neighbours at the earlier layers. Rethinking the feature extraction framework in ViTs, the module for feature representation is the multi-layer perceptron (MLP) and its receptive field can be seen as only itself. So ViTs depend on the multi-head self-attention (MHSA) module to model and capture the relation between tokens. As is pointed out in work [15], with less training data, lower attention layers do not learn to attend locally. In other words, they do not focus on neighbouring tokens and aggregate local information in early stage. As is known, capturing local features in lower layers facilitates the whole representation pipeline. The deep layers sequentially process the low-level texture feature into high-level semantic features for final recognition. Thus ViTs have an extreme performance gap compared with CNNs when training from scratch on small datasets. **On channel aspect**, feature representation exhibits diversity in different channels. And ViT has its own inductive bias that different channel group encodes different feature representation of the object, and the whole token vector forms the representation of the object. As is pointed out in work [16], large datasets tend to help ViT learn strong representation. The insufficient data can not enable ViTs to learn strong enough representation, thus the whole representation is poor for accurate classification.

In this paper, we solve the performance gap of training from scratch on small datasets between CNNs and ViTs and provide a hybrid architecture called Dynamic Hybrid Vision Transformer (DHVT) for substitute. We first introduce a hybrid model to address the issue **on spatial aspect**. The proposed hybrid model integrates a sequence of convolution layers in patch embedding stage to eliminate non-overlapping problem, preserving fine-grained low-level feature, and it involves depth-wise convolution [24] in MLP for local feature extraction. In addition, we design two modules for making feature representation stronger to solve the problem **on channel view**. To be specific, in MLP, depth-wise convolution is adopted for the patch tokens, and the class token is identically passed through without any computation. We then leverage the output patch tokens to produce channel weight like Squeeze-Excitation (SE) [25] for the class token. This operation helps re-calibrate each channel for the class token to reinforce its feature representation. Moreover, in order to enhance interaction among different semantic representation of different channel group and owing to the variable length of token sequence in vision transformer structure, we devise a brand new token mechanism called "head token". The number of head tokens is the same as the number of attention heads in MHSA. Head tokens are generated by segmenting and projecting input tokens along channel. The head tokens will be concatenated with all other tokens to pass through the MHSA. Each channel group in corresponding attention head in the MHSA now is able to interact with others. Though maybe the representation in each channel and channel group is poor for classification on account of insufficient training data, the head tokens help re-calibrate each learned feature pattern and enable a stronger integral representation of the object, which is beneficial to final recognition.

We conduct experiments of training from scratch on various small datasets, the common dataset CIFAR-100 and small domain datasets Clipart, Painting, Sketch from DomainNet [26] to examine the performance of our model. On CIFAR-100, our proposed models show significant performance margin with strong CNNs like ResNeXt, DenseNet and Res2Net. The Tiny model achieves 83.54% with only 5.8M parameters, and our Small model reaches the state-of-the-art 85.68% accuracy with

only 22.8M parameters, outperforming a series of strong CNNs. Therefore, we eliminate the gap between CNNs and ViTs, providing an alternative architecture which can train from scratch on small datasets. We also evaluate the performance of DHVT when training from scratch on ImageNet-1K. Our proposed DHVT-S achieves competitive 82.3% accuracy with only 24.0M parameters, which is the state-of-the-art non-hierarchical vision transformer structure as far as we know, demonstrating the effectiveness of our model on larger datasets. In summary, our main contributions are:

1. We conclude that the data-efficiency on small datasets can be addressed by strengthening two inductive biases in ViTs, which are spatial relevance and diverse channel representation.

2. On spatial aspect, we adopt a hybrid model integrated with convolution, preserving fine-grained low-level feature at earlier stage and forcing the model to extract tokens feature and corresponding neighbour feature.

3. On channel aspect, we leverage the output patch tokens to re-calibrate class token channel-wise, producing better feature representation. We further introduce "head token", a novel design which helps fusing diverse feature representation encoded in different channel group into a stronger integral representation.

2 Related Work

Vision Transformers. Convolutional Neural Networks [27, 28, 29, 30, 31, 32] dominated the computer vision fields in the past decade, with its intrinsic inductive biases designed for image recognition. The past two years witnessed the rise of Vision Transformer models in various vision tasks [33, 11, 9, 6, 34, 35]. Although there exist previous works introducing attention mechanism into CNNs [25, 36, 37], the pioneering full transformer architecture in computer vision are iGPT [38] and ViT [1]. ViT is widely adopted as the architecture paradigm for vision tasks especially image recognition. It processes image as token sequence and exploits relation among tokens. It uses "class token" like BERT [39] to exchange information every layer and for final classification. It performs well when pre-trained on huge datasets. But when training from scratch on ImageNet-1K only, it underperforms ResNets, demonstrating a data-hungry problem.

Data-efficient ViTs. Many of the subsequent modifications on ViT strive for a more data-efficient architecture which can perform well without pre-training on larger datasets. The methods can be divided into different groups. [33, 40] use knowledge distillation strategy and stronger data-augmentation methods to enable training from scratch. [41] points out that using convolution in the patch embedding stage greatly benefits ViTs training. [42, 2, 5, 43, 44] leverage convolution for patch embedding to eliminate the discontinuity brought by non-overlapping patch embedding in vanilla ViT, and such design becomes a paradigm in subsequent works. To further introduce inductive bias into ViT, [2, 18, 22, 45, 46] integrate depth-wise convolution into feed forward network, resulting a hybrid architecture combining the self-attention and convolution. To make ViTs more similar to standard CNNs, [3, 44, 4, 22, 21, 23, 47, 20] re-design the spatial and channel dimension of vanilla ViT, producing a series of hierarchical style vision transformer. [19, 48, 49] design another parallel convolution branch and enable the interaction with self-attention branch, making the two branch complements each other. The above architectures introduce strong inductive bias and become data-efficient when training from scratch on ImageNet-1K. In addition, works like [50, 51, 52] investigate channel-wise representation through conducting self-attention channel-wise, while we enhance channel representation by dynamically aggregating patch token features to enhance class token channel-wise and compatibly involve channel group-wise head tokens into vanilla self-attention. Finally, works like [53, 54, 55], suggesting that the number of tokens can be variable.

ViTs for small datasets. There exists several works on solving the training from scratch problem on small datasets. Though the above modified vision transformers perform well when trained on ImageNet-1K, they fail to compete with standard CNNs when training on much smaller datasets like CIFAR-100. Work [56] introduces a self-supervised style training strategy and a loss function to help training ViTs on small datasets. CCT [57] adopts a convolutional tokenization module and replaces the class token with final sequence pooling operation. SL-ViT [58] adopts shifted patch tokenization module and modifies self-attention to make it focus more locally. Though the previous works reduce the performance gap between standard CNNs ResNets[29], they fail to be sub-optimal when compared with strong CNNs. Our proposed method leverage local constraints and enhance representation interaction, successfully bridging the performance gap on small datasets.

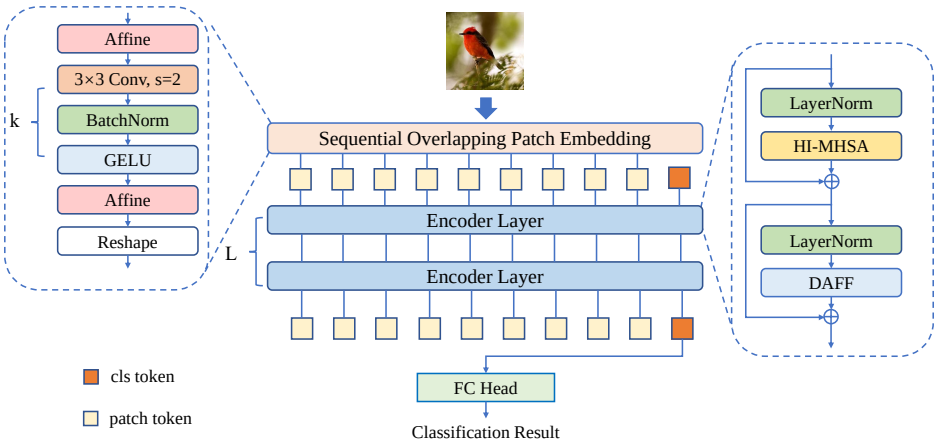

Figure 1: Overview of the proposed Dynamic Hybrid Vision Transformer (DHVT). DHVT follows a non-hierarchical structure, where each encoder layer contains two pre-norm and shortcut, a Head-Interacted Multi-Head Self-Attention (HI-MHSA) and a Dynamic Aggregation Feed Forward (DAFF).

3 Methods

3.1 Overview of DHVT

As shown in Fig. 1, the framework of our proposed DHVT is similar to vanilla ViT. We choose non-hierarchical structure, where every encoder block shares the same parameter setting, processing the same shape of features. Under this structure, we can deal with variable length of token sequence. We keep the design of using class token to interact with all the patch tokens and for final prediction. In the patch embedding module, input image will be split into patches first. Given the input image with resolution $H \times W$ and the target patch size P, the resulting length of patch token sequence will be $N = HW/P^2$. Our modified patch embedding is called Sequential Overlapping Patch Embedding (SOPE), which contains several successive convolution layers of 3×3 convolution with stride $s = 2$, Batch Normalization and GELU [59] activation. The relation between the number of convolution layer and the patch size is $P = 2^k$. SOPE is able to eliminate the discontinuity brought by vanilla patch embedding module, preserving important low-level features. It is able to provide position information to some extent. We also adopt two affine transformations before and after the series of convolution layers. This operation rescales and shifts the input feature, and it acts like normalization, making the training performance more stable on small datasets. The whole process of SOPE can be formulated as follows.

$$Aff(\mathbf{x}) = Diag(\boldsymbol{\alpha})\mathbf{x} + \boldsymbol{\beta} \tag{1}$$

$$G_i(\mathbf{x}) = GELU(BN(Conv(\mathbf{x}))), i = 1, \ldots, k \tag{2}$$

$$SOPE(\mathbf{x}) = Reshape(Aff(G_k(\ldots(G_2(G_1(Aff(\mathbf{x})))))))) \tag{3}$$

In Eq.1, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are learnable parameters, and initialized as 1 and 0 respectively. After the sequence of convolution layers, the feature maps are then reshaped as patch tokens and concatenated with a class token. Then the sequence of token will be fed into encoder layers. After SOPE, token sequence will pass through layers of encoder, where each encoder contains Layer Normalization [60], multi-head self-attention and feed forward network. Here we modified the MHSA as Head-Interacted Multi-Head Self-Attention (HI-MHSA) and feed forward network as Dynamic Aggregation Feed Forward (DAFF). We will introduce them in the following sections. After the final encoder layer, the output class token will be fed into linear head for final prediction.

3.2 Dynamic Aggregation Feed Forward

The vanilla feed forward network (FFN) in ViT is formed by two fully-connected layers and GELU activation. All the tokens, either patch tokens or class token, will be processed by FFN. Here

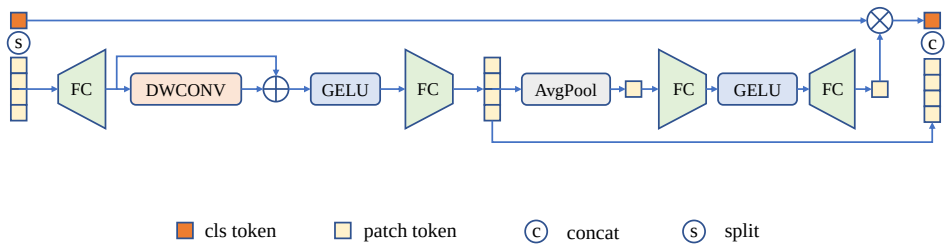

Figure 2: The structure of Dynamic Aggregation Feed Forward (DAFF).

we integrate depth-wise convolution [24] (DWConv) in FFN and resulting a hybrid model. Such hybrid model is similar to standard CNNs because it can be seen as using convolution to do feature representation. With the inductive bias brought by depth-wise convolution, the model is forced to capture neighbouring feature, solving the problem on spatial view. It greatly reduces the performance gap when training from scratch on small datasets, and converges faster than standard CNNs. However, such structure still performs worse than stronger CNNs. More investigations are required to solve the problem on channel aspect.

We propose two methods that make the whole model more dynamic and learn stronger feature representation under insufficient data. The first proposed module is Dynamic Aggregation Feed Forward (DAFF). We aggregate the feature of patch tokens into class token in an channel attention way, similar to Squeeze-Excitation operation in SENet [25], as is shown in Fig. 2. Class token is split before the projection layers. Then the patch tokens will go through a depth-wise integrated multi-layer perceptron with shortcut inside. The output patch tokens will then be averaged into a weight vector $\mathbf{W}$. After the squeeze-excitation operation, the output weight vector will be multiplied with class token channel-wise. Then the re-calibrated class token will be concatenated with output patch tokens to restore the token sequence. We use $\mathbf{X}_c, \mathbf{X}_p$ to denote class token and patch tokens respectively. The process can be formulated as:

$$\mathbf{W} = Linear(GELU(Linear((Average(\mathbf{X}_p)))) \tag{4}$$

$$\mathbf{X}_c = \mathbf{X}_c \odot \mathbf{W} \tag{5}$$

3.3 Head Token

The second design to enhance feature representation is "head token", which is a brand new mechanism as far as we know. There are two reasons why we introduce head token here. First, in the original MHSA module, each attention head is not interacted with others, which means each head only focus on itself to calculate attention. Second, channel groups in different head are responsible for different feature representation, which is the inductive bias of ViTs. And as we pointed out above, the lack of training data can not enable models to learn strong representation. Under this circumstance, the representation in each channel group is too weak for recognition. After introducing head tokens into attention calculation, the channel group in each head are able to interact with those in other heads, and different representation can be fused into an integral representation of the object. Representation learned by insufficient data may be poor in each channel, but their combination will produce an strong enough representation. The structure of vision transformer also guarantees this mechanism because the length of input tokens is variable, except for the hierarchical structure vision transformer with window attention such as[4, 23].

The process of generating head tokens are shown as Fig. 3 (a). We denote the number of patch tokens as N, so the length of input sequence is $N + 1$. According to the pre-defined number of heads h, each D-dimensional token, including class token, will be reshaped into h parts. Each part contains d channels, where $D = d \times h$. We average all the separated tokens in their own parts. Thus we get totally h tokens and each one is d-dimensional. All such intermediate tokens will be projected into D-dimension again, resulting h head tokens in total. The head tokens will be added with head embedding, which provides positional information for head tokens. Head embedding is a group of learnable parameters, just like positional embedding. Finally, they are concatenated with patch tokens and class token, forming the token sequence for standard MHSA, as Eq. 7, in which $\mathbf{X}_H$ denotes

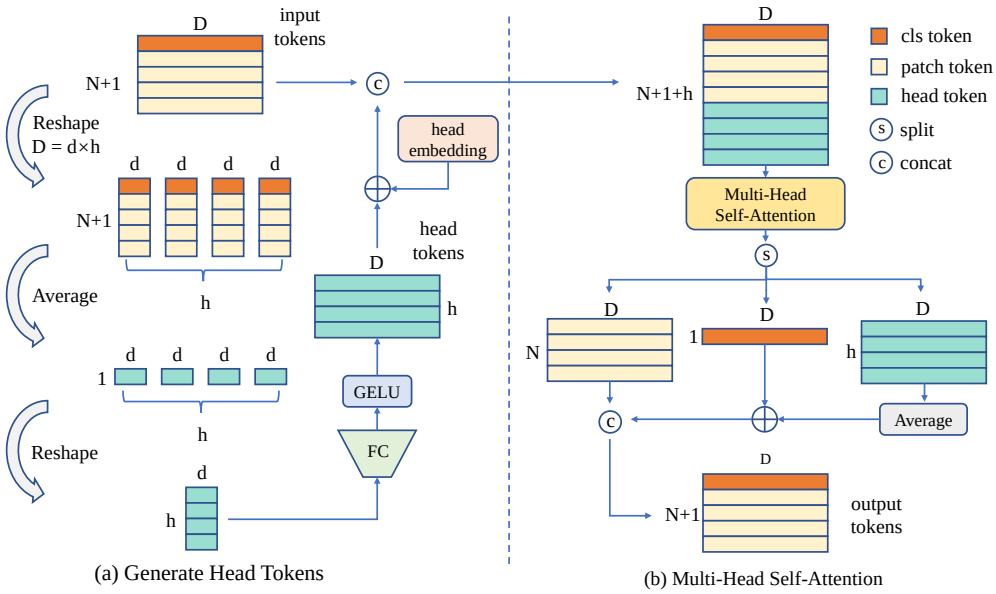

Figure 3: Pipeline of Head-Interacted Multi-Head Self-Attention (HI-MHSA).

head tokens. We do not change the attention calculation in MHSA. Head tokens will also be linearly projected into query, key and value, and they will interacted with all other tokens. After MHSA, the head tokens will be averaged and added to class token, just as Fig. 3 (b) shows. Head tokens can be derived as Eq. 6 shows. We use $\mathbf{E}_{head}$ to denote head embedding.

$$\mathbf{X}_H = GELU(Linear((Average(Reshape(\mathbf{X}))))) + \mathbf{E}_{head} \tag{6}$$

$$\mathbf{X} = [\mathbf{X}_c; \mathbf{X}_p; \mathbf{X}_H] = [\mathbf{X}_c; \mathbf{X}_p^1, \ldots, \mathbf{X}_p^N; \mathbf{X}_H^1, \ldots, \mathbf{X}_H^h] \tag{7}$$

4 Experiments

All the experiments presented in our paper are based on image classification. We do not conduct experiments on downstream tasks. We first introduce the training datasets and experimental settings in Section 4.1. The performance comparisons are shown in Section 4.2. We also show the result of ablation study in Section 4.3. And finally we present an example of visualization in 4.4.

4.1 Datasets and Experimental Settings

Datasets. Our main focus is training from scratch on small datasets. There are two factors to consider whether a dataset is small: the total number of training data in the dataset and the average number of training data for each class. Some datasets are small on the first factor, but large on the second. The example is CIFAR-10 [61], with 50000 training data in total for 10 classes, has an average of 5000 instances in each class. Considering this, we do not choose CIFAR-10 as our target dataset here. We choose 5 different datasets here. The main performance comparisons are on CIFAR-100 [61]. And we choose three datasets from DomainNet [26], a benchmark commonly for domain adaptation tasks. They have a large domain-shift from common medium dataset ImageNet-1K [14], making the fine-tunning experiments non-trivial, as pointed in [56]. Finally, we also choose ImageNet-1K to test the performance of our proposed model. The details of the datasets are shown in Table 1.

Model Variants. We propose two architecture variants.

- DHVT-T: 12 encoder layers, embedding dimension of 192, MLP ratios of 4, attention heads of 4 on CIFAR-100 and DomainNet, and 3 on ImageNet-1K.
- DHVT-S: 12 encoder layers, embedding dimension of 384, MLP ratios of 4, attention heads of 8 on CIFAR-100, 6 on DomainNet and ImageNet-1K.

Table 1: The details of training datasets. We report the train and test size of each dataset, including the number of class. We also show the average images per class in the training set.

Dataset	Train size	Test size	Classes	Average images per class
CIFAR-100 [61]	50000	10000	100	500
ClipArt [26]	33525	14604	345	97
Sketch [26]	48212	20916	345	140
Painting [26]	50416	21850	345	146
ImageNet-1K [14]	1281167	100000	1000	1281

Implementation Details. When training our DHVT, we keep the image size in CIFAR-100 as its original resolution 32×32, and patch size is set to 4 or 2. For ImageNet-1K, ClipArt, Painting and Sketch, we adopt resolution 224×224, and the patch size comes to 16. All the data-augmentations are the same as those in DeiT [33]. We do not tune data-augmentation hyperparameters for better performance. On all of the datasets, we train our network from random initialization with the AdamW [62] optimizer with a cosine decay learning-rate scheduler. We set batch size of 512 and 256 for DHVT-T and DHVT-S when training on CIFAR-100, an initial learning rate of 0.001, and a weight decay of 0.05, warm-up epoch of 5. When on ClipArt, Sketch and Painting, we use batch size of 256 and 128 respectively for DHVT-T and DHVT-S, the initial learning rate of 0.001, warm-up epoch of 20 and weight decay of 0.05. For ImageNet-1K, we use the batch size of 512 for both models and initial learning rate of 0.0005 and weight decay of 0.05, warm-up epoch of 10. All of the training devices are Nvidia 3090 GPUs. We use Pytorch tools and our code is modified from timm[1].

Table 2: Performance comparison of different method on CIFAR-100 dataset. All models are trained from random initialization.

Type	Method	Params	Patch Size	Epochs	Accuracy (%)
CNN	WRN28-10 [63]	36.5M	1	200	80.75
	SENet-29 [25]	35.0M	1	300	82.22
	ResNeXt-29, 8×64d [30]	34.4M	1	300	82.23
	SKNet-29 [32]	27.7M	1	300	82.67
	DenseNet-BC (k = 40) [31]	25.6M	1	300	82.82
	Res2NeXt-29, 6c×24w×6s-SE [64]	36.9M	1	300	83.44
ViT	DeiT-T [23]	5.3M	2	300	67.52
	DeiT-S [23]	21.3M	2	300	69.78
	PVT-T [23]	12.8M	1	300	69.62
	PVT-S [23]	24.1M	1	300	69.79
	Swin-T [23]	27.5M	1	300	78.07
	NesT-T [23]	6.2M	1	300	78.69
	NesT-S [23]	23.4M	1	300	81.70
	NesT-B [23]	90.1M	1	300	82.56
Hybrid	CCT-7/3×1 [57]	3.7M	4	300	80.92
	DHVT-T (Ours)	6.0M	4	300	80.93
	DHVT-S (Ours)	23.4M	4	300	82.91
	DHVT-T (Ours)	**5.8M**	2	300	**83.54**
	DHVT-S (Ours)	**22.8M**	2	300	**85.68**

4.2 Performance Comparisons

Results on CIFAR-100. We mainly compare the performance of our proposed model on CIFAR-100. Patch size set to 1 means taking raw pixel input. For comparison of other methods, we directly cite the results reported in the corresponding paper. The results of our model are the best out of five runs with different random seed. As is shown in Table 2, our model DHVT-T reaches 83.54 with only 5.8M parameters. and DHVT-S reaches 85.68 with only 22.8M parameters. With much less

[1] https://github.com/rwightman/pytorch-image-models

parameters, our model has a much higher performance against other ViT based models and strong CNNs ResNeXt, SENet, SKNet, DenseNet and Res2Net. We not only bridge the performance gap between CNNs and ViTs, but also push the state-of-the-art result to a higher level.

Table 3: Results on DomainNet

Method	Params	ClipArt	Painting	Sketch
ResNet-50	24.2M	71.90	64.36	67.45
DHVT-T	6.1M	71.73	63.34	66.60
DHVT-S	23.8M	**73.89**	**66.08**	**68.72**

Table 4: Results on ImageNet-1K

Method	Params	ImageNet-1K
DHVT-T	6.2M	76.5
DHVT-S	24.0M	82.3

Results on DomainNet. We also conduct experiments on other small datasets. Here we choose three datasets from DomainNet as our target. We use the implementation of ResNet-50 in Pytorch official code for performance comparison. All of the data-augmentations, such as Mixup [65] and CutMix [66] and AutoAugment [67], are also adopted for training ResNet-50 from scratch on these datasets. All of the results reported are the best out of four runs. As is shown in Table 3, our model shows better results than standard ResNet-50, demonstrating its performance across different small datasets.

Results on ImageNet-1K To test the train-from-scratch performance of our model on common medium size dataset ImageNet-1K, we also conduct experiments on it. We follow the same experimental settings as in DeiT [33]. The results are shown in the Table 4. Surprisingly, our DHVT-T reaches 76.47 accuracy and our DHVT-S reaches 82.3 accuracy. As far as we know, this is the best performance under such non-hierarchical vision transformer structure with class token. And our model outperforms many of the state-of-the-art methods with comparable parameters. This experiment shows that our model not only behaves well on small datasets, but also exhibits powerful performance on larger datasets. We will show the performance comparison with other methods training from scrach on ImageNet-1K in the supplementary materials.

4.3 Ablation Studies

All the results in the ablation study are the average over four runs with different random seed. The model for ablation study is DHVT-T, with patch size of 4 and training from scratch on CIFAR-100 with same data-augmentation as in Section 4.2. Here DHVT-T is trained with learning rate of 0.001, warm-up epoch of 10 and batch size of 512, total epoch of 300. The baseline is DeiT-T with 4 heads and the patch size is set to 4. The results are shown in the following tables.

Table 5: Ablation study on SOPE and DAFF

Abs. PE	SOPE	DAFF	Acc
✓	✗	✗	67.59 (+0.00)
✗	✗	✗	58.72 (-8.87)
✓	✓	✗	73.68 (+6.09)
✗	✓	✗	69.65 (+2.06)
✓	✗	✓	79.47 (+11.88)
✗	✗	✓	79.75 (+12.16)
✓	✓	✓	80.17 (+12.58)
✗	✓	✓	80.35 (+12.76)

Table 6: Ablation study on head token

Abs. PE	SOPE & DAFF	Head Token	Acc
✓	✗	✗	67.59 (+0.00)
✓	✗	✓	69.10 (+1.51)
✗	✓	✓	80.85 (+13.26)

The importance of positional information. We have baseline performance 67.59 from DeiT-T with 4 heads, training from scratch with 300 epochs. When removing absolute positional embedding, the performance drops drastically to 58.72, demonstrating the importance of position information in vision transformers. SOPE is able to provide positional information to some extent because such absolute positional information can be derived from the zero padding. As is shown in Table 5, when adopting SOPE and removing absolute position embedding, the performance does not drop so drastically. But only depending on SOPE to provide position information is not enough.

The role of DAFF. When adopting DAFF, the performance gain increases greatly to 79.47, because DAFF solve the problem on both spatial and channel aspect, introducing strong local constraints and

re-calibrating channel feature representation. It is sensible to see that removing absolute position embedding can increase performance. The positional information have been encode into tokens through the depth-wise convolution in DAFF, and the absolute position embedding will break translation invariance. When both SOPE and DAFF are adopted, the positional information will be encoded comprehensively, and SOPE will also help address non-overlapping problem here, preserving fine-grained low-level feature in early stage.

The role of head tokens. From Table 6, we can also see the stable performance gain brought by head tokens across different model structure. When introducing head tokens into DeiT-T, the performance gets a +1.51 gain, demonstrating its effectiveness. As we said before, head tokens guarantee the interaction among different channel groups, better fusing the diverse representation. The resulting integral representation is now strong enough for classification. When adopting all three modifications, we get +13.26 accuracy gain, successfully bridging the performance gap with CNNs.

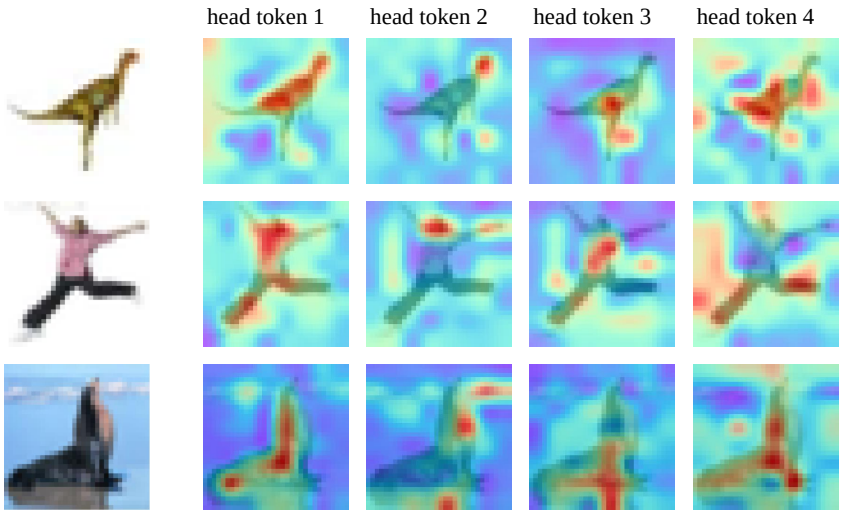

Figure 4: Visualization of the attention map of head tokens to patch tokens on low layer

4.4 Visualization

We visualize the attention maps of head tokens to patch tokens in Fig. 4. Each row represents one image. The results are samples in the second encoder layer. We can see that different head token activates on different patch tokens, exhibiting its diverse representation. On such low layer, low-level fine-grained feature is able to be captured in our model. More visualization results are shown in the supplementary materials.

4.5 Limitation

Though we achieve a much higher performance than existing methods, such performance gain comes at the expense of computation. The performance when patch size set to 2 boosts higher than using patch size of 4. But the computation expense rises quadratically. In practical usage, we suggest choose a good patch size for better trade-off between performance and computation. We will show the FLOPs and throughput in the supplementary materials.

5 Conclusion

In this paper, we present an alternative vision transformer architecture DHVT, which can train from scratch on small datasets and reach state-of-the-art performance on series of datasets. The weak inductive biases of spatial relevance and diverse channel representation brought by insufficient training data is strengthened in our model. The highlighted head token design is able to transferred to variants of ViT model to enable better feature representation.

References

[1] Dosovitskiy, A., L. Beyer, A. Kolesnikov, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[2] Yuan, K., S. Guo, Z. Liu, et al. Incorporating convolution designs into visual transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 579–588. 2021.

[3] Wang, W., E. Xie, X. Li, et al. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 568–578. 2021.

[4] Liu, Z., Y. Lin, Y. Cao, et al. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022. 2021.

[5] Wu, H., B. Xiao, N. Codella, et al. Cvt: Introducing convolutions to vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22–31. 2021.

[6] Carion, N., F. Massa, G. Synnaeve, et al. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[7] Dai, Z., B. Cai, Y. Lin, et al. Up-detr: Unsupervised pre-training for object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1601–1610. 2021.

[8] Zhu, X., W. Su, L. Lu, et al. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[9] Strudel, R., R. Garcia, I. Laptev, et al. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7262–7272. 2021.

[10] Guo, R., D. Niu, L. Qu, et al. Sotr: Segmenting objects with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7157–7166. 2021.

[11] Jiang, Y., S. Chang, Z. Wang. Transgan: Two pure transformers can make one strong gan, and that can scale up. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] Hudson, D. A., L. Zitnick. Generative adversarial transformers. In *International Conference on Machine Learning*, pages 4487–4499. PMLR, 2021.

[13] Vaswani, A., N. Shazeer, N. Parmar, et al. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[14] Russakovsky, O., J. Deng, H. Su, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[15] Raghu, M., T. Unterthiner, S. Kornblith, et al. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34, 2021.

[16] Park, N., S. Kim. How do vision transformers work? *arXiv preprint arXiv:2202.06709*, 2022.

[17] d'Ascoli, S., H. Touvron, M. L. Leavitt, et al. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021.

[18] Li, Y., K. Zhang, J. Cao, et al. Localvit: Bringing locality to vision transformers. *arXiv preprint arXiv:2104.05707*, 2021.

[19] Peng, Z., W. Huang, S. Gu, et al. Conformer: Local features coupling global representations for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 367–376. 2021.

[20] Chen, Z., L. Xie, J. Niu, et al. Visformer: The vision-friendly transformer. *CoRR*, abs/2104.12533, 2021.

[21] Zhao, Y., G. Wang, C. Tang, et al. A battle of network structures: An empirical study of cnn, transformer, and mlp. *arXiv preprint arXiv:2108.13002*, 2021.

[22] Heo, B., S. Yun, D. Han, et al. Rethinking spatial dimensions of vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11936–11945. 2021.

[23] Zhang, Z., H. Zhang, L. Zhao, et al. Nested hierarchical transformer: Towards accurate, data-efficient and interpretable visual understanding. In *AAAI Conference on Artificial Intelligence (AAAI), 2022*. 2022.

[24] Howard, A. G., M. Zhu, B. Chen, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[25] Hu, J., L. Shen, G. Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141. 2018.

[26] Peng, X., Q. Bai, X. Xia, et al. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415. 2019.

[27] Krizhevsky, A., I. Sutskever, G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[28] Szegedy, C., W. Liu, Y. Jia, et al. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9. 2015.

[29] He, K., X. Zhang, S. Ren, et al. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778. 2016.

[30] Xie, S., R. Girshick, P. Dollár, et al. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500. 2017.

[31] Huang, G., Z. Liu, L. Van Der Maaten, et al. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708. 2017.

[32] Li, X., W. Wang, X. Hu, et al. Selective kernel networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 510–519. 2019.

[33] Touvron, H., M. Cord, M. Douze, et al. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[34] Arnab, A., M. Dehghani, G. Heigold, et al. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6836–6846. 2021.

[35] Chen, X., S. Xie, K. He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9640–9649. 2021.

[36] Wang, X., R. Girshick, A. Gupta, et al. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803. 2018.

[37] Hu, H., Z. Zhang, Z. Xie, et al. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473. 2019.

[38] Chen, M., A. Radford, R. Child, et al. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020.

[39] Devlin, J., M.-W. Chang, K. Lee, et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[40] Jiang, Z.-H., Q. Hou, L. Yuan, et al. All tokens matter: Token labeling for training better vision transformers. *Advances in Neural Information Processing Systems*, 34, 2021.

[41] Xiao, T., M. Singh, E. Mintun, et al. Early convolutions help transformers see better. *Advances in Neural Information Processing Systems*, 34:30392–30400, 2021.

[42] Yuan, L., Y. Chen, T. Wang, et al. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 558–567. 2021.

[43] Chu, X., Z. Tian, B. Zhang, et al. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[44] Wang, W., E. Xie, X. Li, et al. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, pages 1–10, 2022.

[45] Ren, S., D. Zhou, S. He, et al. Shunted self-attention via multi-scale token aggregation, 2021.

[46] Guo, J., K. Han, H. Wu, et al. Cmt: Convolutional neural networks meet vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12175–12185. 2022.

[47] Mao, X., G. Qi, Y. Chen, et al. Towards robust vision transformer. *arXiv preprint arXiv:2105.07926*, 2021.

[48] Chen, Q., Q. Wu, J. Wang, et al. Mixformer: Mixing features across windows and dimensions. *arXiv preprint arXiv:2204.02557*, 2022.

[49] Xu, Y., Q. Zhang, J. Zhang, et al. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems*, 34, 2021.

[50] Ali, A., H. Touvron, M. Caron, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34, 2021.

[51] Ding, M., B. Xiao, N. Codella, et al. Davit: Dual attention vision transformer. *arXiv preprint arXiv:2204.03645*, 2022.

[52] Xu, W., Y. Xu, T. Chang, et al. Co-scale conv-attentional image transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9981–9990. 2021.

[53] Ryoo, M., A. Piergiovanni, A. Arnab, et al. Tokenlearner: Adaptive space-time tokenization for videos. *Advances in Neural Information Processing Systems*, 34, 2021.

[54] Fang, J., L. Xie, X. Wang, et al. Msg-transformer: Exchanging local spatial information by manipulating messenger tokens. In *CVPR*. 2022.

[55] Liang, Y., C. Ge, Z. Tong, et al. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*. 2022.

[56] Liu, Y., E. Sangineto, W. Bi, et al. Efficient training of visual transformers with small datasets. *Advances in Neural Information Processing Systems*, 34, 2021.

[57] Hassani, A., S. Walton, N. Shah, et al. Escaping the big data paradigm with compact transformers. *arXiv preprint arXiv:2104.05704*, 2021.

[58] Lee, S. H., S. Lee, B. C. Song. Vision transformer for small-size datasets, 2021.

[59] Hendrycks, D., K. Gimpel. Gaussian error linear units (gelus), 2016.

[60] Ba, J. L., J. R. Kiros, G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[61] Krizhevsky, A., G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[62] Loshchilov, I., F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[63] Zagoruyko, S., N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[64] Gao, S.-H., M.-M. Cheng, K. Zhao, et al. Res2net: A new multi-scale backbone architecture. *IEEE transactions on pattern analysis and machine intelligence*, 43(2):652–662, 2019.

[65] Zhang, H., M. Cisse, Y. N. Dauphin, et al. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*. 2018.

[66] Yun, S., D. Han, S. J. Oh, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032. 2019.

[67] Cubuk, E. D., B. Zoph, D. Mane, et al. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123. 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [Yes]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes]

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.