A Experiment on CIFAR100

We have run additional experiments on CIFAR100 following the setup in DeiT [2] but w/o pretraining on ImageNet1K.

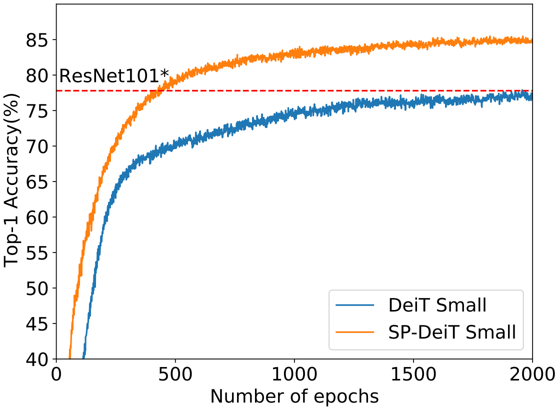
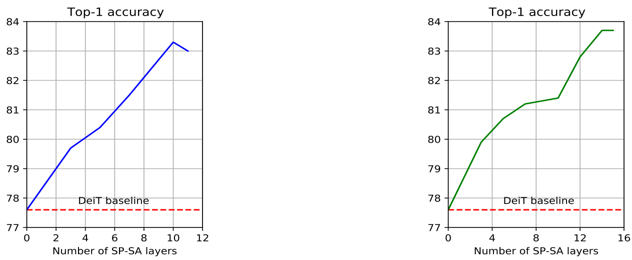


Figure 1: Training SP-ViT (DeiT [2] as baseline) on CIFAR100.

B Numbers of Substituted SA Layers



(a) 12 layer SP-ViT

(b) 16 layer SP-ViT

Figure 2: Accuracy(%) of SP-ViT on ImageNet-100 with different numbers of SP-SA layers. Fig. 2a and Fig. 2b show consistent improvements of our SP-ViT over DeiT Baselines with a total number of 12 and 16 layers respectively.

We first investigate how the model performance is affected by the number of SP-SA layers. The layers are substituted from bottom to top and a classification token is inserted after the last SP-SA layer. It is shown in Fig. 2a that substituting a number of SA layers with SP-SA results in improved accuracy comparing to DeiT baseline (0 layer). In general, the performance improves as more layers are substituted. For a model with 12 layers, the best performance is achieved when 10 layers are substituted. When substituting all but the last SA layer with SP-SA, the performance drops slightly. We hypothesize that when the classification token is only involved in the last layer, the class-specific features are not adequately

Sub. layers	Cls token insertion layers	Top-1 (%)
0	0	77.6
0	10	81.7
10	10	**83.3**
0	Global Average Pooling	79.5
12	Global Average Pooling	**81.7**

Table 1: Eliminate the effect of inserting the class token at later layers on ImageNet-100.

extracted. We further investigated a deeper model in Fig. 2b, and found the similar trend. The best performance is achieved when the first to the penultimate layer are substituted. As discussed in the main text, we add the classification token directly after SP-SA layers because it has no valid 2D relative coordinate. To exclude the influence of inserting it at deeper layers instead of the first, we conduct a further comparison in Tab. 1.

C More Experiment Details

We show in Tab. 2 the default hyperparameters for training our SP-ViT on ImageNet-1K based on DeiT and LV-ViT respectively. All hyperparameter settings follow the baselines' except that for DeiT-based SP-ViTs we adopt a smaller learning rate.

Base Config.	DeiT	LV-ViT
Supervision	Standard	Token labeling
SP-SA layers	10	10
Epoch	300	300
Optimizer	AdamW	AdamW
Batch size	1024	1024
LR	$2.5e-4 \cdot \frac{\text{batch size}}{512}$	$1e-3 \cdot \frac{\text{batch size}}{640}$
LR decay	cosine	cosine
Weight decay	0.05	0.05
Warmup epochs	5	5
Label smoothing ε	0.1	0.1
Stoch. Depth	0.1	0.1
Repeated Aug	✓	-
RandAug	9/0.5	9/0.5
Mixup prob.	0.8	-
Erasing prob.	0.25	0.25

Table 2: Default hyperparameters for our SP-ViTs on ImageNet-1K.

For our SP-ViT trained on ImageNet-1K, we further adopt the Conditional Positional Encoding (CPE) [1], which is found to be effective as shown in Tab. 3.

Model	CPE [1]	Top-1 (%)
SP-ViT-S	-	83.7
	✓	83.9
SP-ViT-M	-	84.7
	✓	84.9
SP-ViT-L	-	85.3
	✓	85.5

Table 3: Effect of Conditional Positional Encoding [1] on ImageNet-1K.

D Python Implementation

We also list our Pytorch implementation of SP-SA List. 1 SP-SA can be easily integrated into any existing vision transformer models by directly replacing a number of SA layers. Calculating the relative coordinates to query patches is trival, so this part of code is not included for simplicity. Note that the insertion of classification token should be moved after SP-SA layers, as mentioned in the main text.

E More Visualization

We provide more examples of learned Spatial Priors (SP) by our SP-ViT based on DeiT-Small and trained on ImageNet-1K in Fig. 3 and Fig. 4.

References

[1] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.

[2] Hugo Touvron, Matthieu Cord, Douze Matthijs, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *ICML 2021: 38th International Conference on Machine Learning*, 2021.

Listing 1 SP-SA `SP-SA.py`

```python
import torch
from torch import nn

class SP_SA(nn.Module):
    def __init__(self, dim, num_heads=8, qk_scale=None, attn_drop=0.,
        proj_drop=0., rel_indices=None, **kwargs):
        super().__init__()
        self.num_heads = num_heads
        self.dim = dim
        head_dim = dim // num_heads
        self.scale = qk_scale or head_dim ** -0.5
        self.v = nn.Linear(dim, dim, bias=False)
        self.qk = nn.Linear(dim, dim * 2, bias=False)
        self.w1 = nn.Linear(2, dim, bias=True)
        self.w2 = nn.Parameter(torch.zeros(dim, 1))
        self.b2 = nn.Parameter(torch.ones(num_heads))

        self.attn_drop = nn.Dropout(attn_drop)
        self.proj = nn.Linear(dim, dim)
        self.proj_drop = nn.Dropout(proj_drop)
        self.act = nn.ReLU()
        self.rel_indices = rel_indices

    def forward(self, x):
        B, N, C = x.shape
        attn = self.get_attention(x)

        v = self.v(x).reshape(B, N, self.num_heads, C // self.num_heads).
            permute(0, 2, 1, 3)
        x = (attn @ v).transpose(1, 2).reshape(B, N, C)
        x = self.proj(x)
        x = self.proj_drop(x)
        return x

    def get_attention(self, x):
        B, N, C = x.shape

        # Calculating Patch Score
        qk = self.qk(x).reshape(B, N, 2, self.num_heads, C // self.
            num_heads).permute(2, 0, 3, 1, 4)
        q, k = qk[0], qk[1]
        patch_score = (q @ k.transpose(-2, -1)) * self.scale

        # Calculating Spatial Prior
        sp_hidden = self.w1(self.rel_indices).view(1, N, N, self.
            num_heads, self.dim // self.num_heads)
        sp = torch.einsum('nm,hijnm->hijn', (self.w2.view(self.num_heads,
             -1), self.act(sp_hidden))) + self.b2
        sp = sp.repeat(B, 1, 1, 1)

        enhanced_attention = (patch_score * sp.permute(0, 3, 1, 2)).
            softmax(dim=-1)
        attn = self.attn_drop(enhanced_attention)
        return attn
```

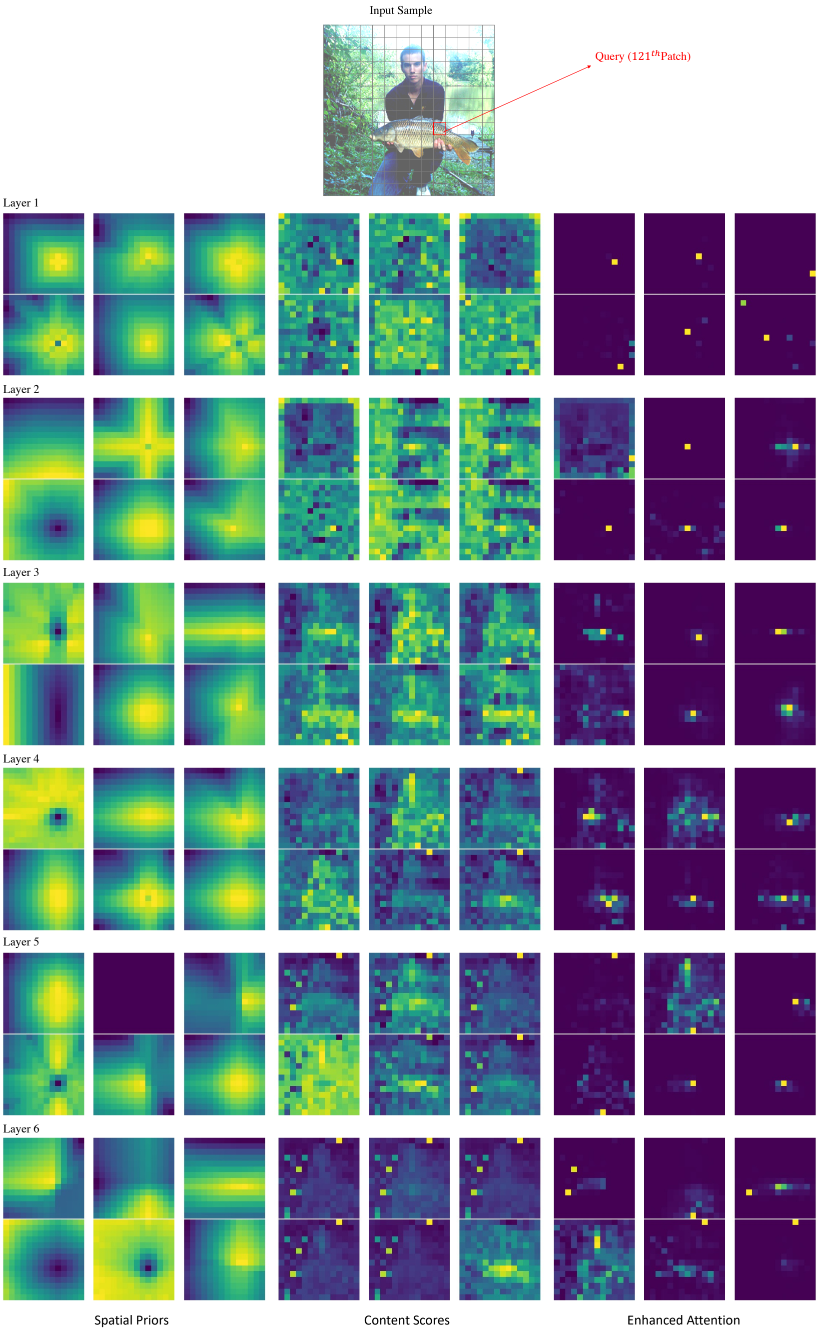


Figure 3: More Visualization of the learned 2D SPs, content scores and the enhanced attention of layer 1-6 for the 121^{th} query patch.

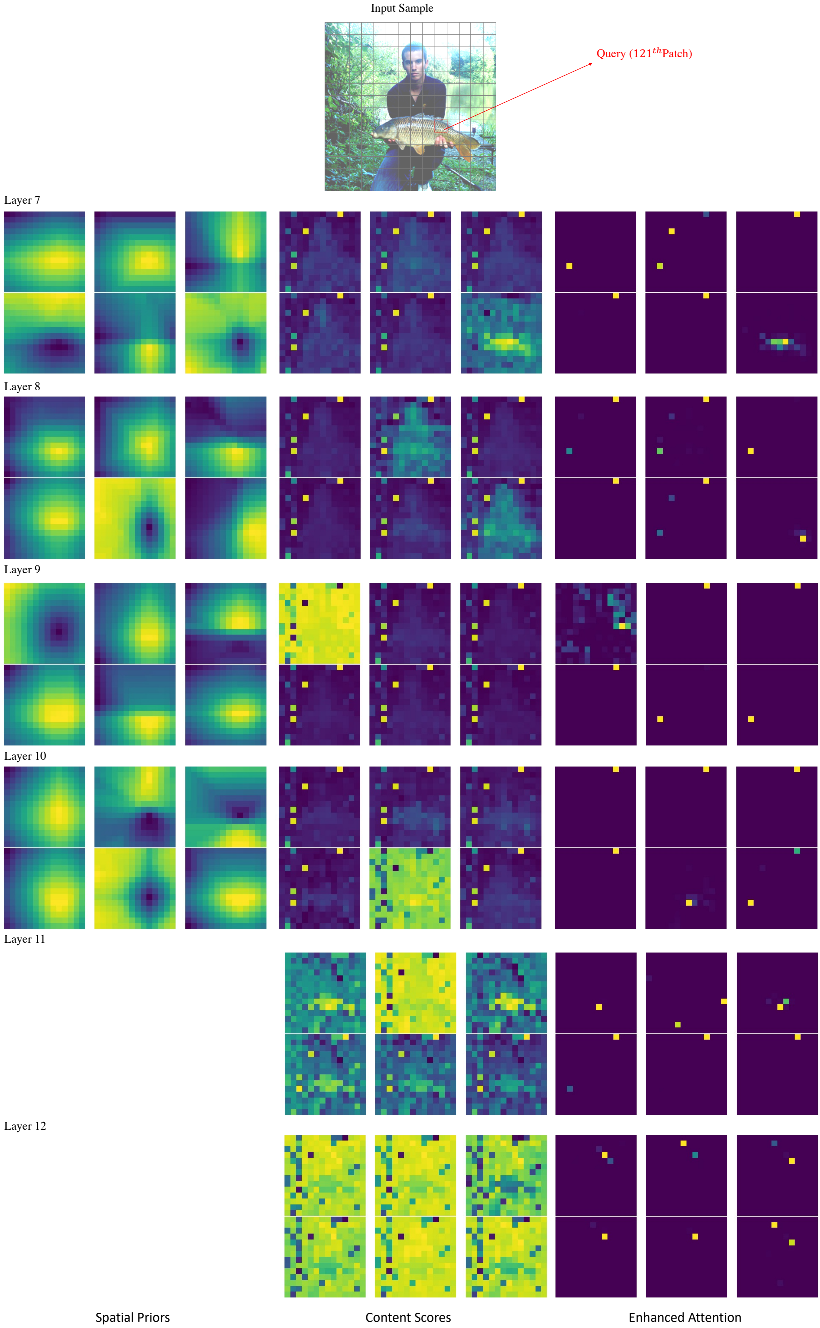


Figure 4: More Visualization of the learned 2D SPs, content scores and the enhanced attention of layer 7-12 for the 121^{th} query patch. Note that layer 11 and 12 are vanilla SA layers, thus no spatial priors are existed.