Appendix

The content of Appendix is summarized as follows: 0) in Sec. A, we discuss the justification of using ViT in few-shot learning scenarios; 1) in Sec. B, we list the network architectures we used in the experiment; 2) in Sec. C, we state implementation and training details to ensure that our SUN can be reproduced; 3) in Sec. D and Sec. E, we introduce the detail of SUN with FEAT (SUN-F) and DeepEMD (SUN-D), then demonstrating its performance; 4) in Sec. F, we conduct more ablation study to analyze other components of SUN; 5) and in Sec. G, we conduct t-SNE visualization to qualitatively evaluate ViT with SUN, thus demonstrating the effectiveness of SUN.

Table 1. Detailed layer specification of the ViTs we used [11,19,38,7] for few-shot classification. All ViTs are scaled to the size with ∼12.5M parameters such that they own approximately same parameter number with the widely-used ResNet-12. Specifically, $k \times k$ means convolution operation with kernel size of k, d means channel dimension, s means stride, "res" means the residual connection from the input via a 3×3 convolution with s=2, MaxPool means max pooling operation, MHSA means self-attention layer, S-MHSA means shifted self-attention layer proposed by [19], FFN means feed-forward layer and C-FFN means FFN with 3×3 depthwise convolution. We strictly follow the corresponding official implementation to build these ViTs. And to simplify the layer description, we omit normalization layers in the table.

Stage	Layers	LV-ViT [11]	Swin Transformer [19]	NesT [38]	Visformer [7]
1	Patch Embedding	3×3, d=96, s=2 3×3, d=96 3×3, d=96 (+res) MaxPool, 2x2 4×4, d=384, s=4	3×3, d=64, s=2 3×3, d=64 3×3, d=144 (+res) MaxPool, 2x2	3×3, d=64, s=2 3×3, d=64 3×3, d=128 (+res) MaxPool, 2x2	3×3, d=32, s=2 3×3, d=32 3×3, d=128 (+res) MaxPool, 2x2
	ViT Blocks	MHSA, d=384 FFN, ratio=3	MHSA, d=144 S-MHSA, d=144 FFN, ratio=4	MHSA, d=384 FFN, ratio=4	MHSA, d=384 C-FFN, ratio=4
	Blocks Num	8	2	2	4
2	Patch Embedding	/	2×2, d=288, s=2	3×3, d=384 MaxPool, 2×2	2×2, d=256, s=2
	ViT Blocks	/	MHSA, d=288 S-MHSA, d=288 FFN, ratio=4	MHSA, d=384 FFN, ratio=4	MHSA, d=256 C-FFN, ratio=4
	Blocks Num	/	3	2	2
3	Patch Embedding	/	2×2, d=576, s=2	3×3, d=512 MaxPool, 2×2	2×2, d=512, s=2
	ViT Blocks	/	MHSA, d=576 S-MHSA, d=576 FFN, ratio=4	MHSA, d=512 FFN, ratio=4	MHSA, d=512 C-FFN, ratio=4
	Blocks Num	/	2	2	3
Parameters (M)		12.6	12.6	12.8	12.5

A Why Using ViT for Few-Shot Learning

The justification of using ViTs for few-shot learning are two-fold: **a)** though ViT cannot well handle few-shot learning as CNN now, it has three adantages over CNN. 1) ViTs often achieve better performance than CNNs of the same model size when training data is at moderate-scale; 2) ViTs exhibit great potential to unify vision and language models, while existing CNNs mainly work well on vision tasks; 3) ViTs are highly parallelized, and can be more ficiently trained and tested than CNNs. So we hope to build a few-shot ViT training framework to release the above powers in few shot learning. Meanwhile, **b)** our SUN framework uses few-shot learning as an example to prove that ViTs can indeed perform well on such scenarios.

B Network Architecture

We evaluate our SUN on four different ViTs, i.e., LV-ViT [11] (standard ViT), Swin Transformer [19] (shifting-window ViT), Visformer [7] (CNN-enhanced ViT) and NesT [38] (locality-enhanced ViT), which cover most of existing ViT types. For a fair comparison, we scale the depth and width of these ViTs such that their model sizes are similar to ResNet-12 [9] ($\sim$12.5M parameters) which is the most commonly used architecture and achieves (nearly) state-of-the-art performance on few-shot classification tasks.

To make our SUN easier to reproduce, we list the network architectures of ViTs we used in this paper. The detailed layer specification of these ViTs is shown as Table . Specifically, the single stage LV-ViT includes a three-layer overlapped patch embedding with residual connection and eight stacking standard transformer encoder blocks. Given input image with 80×80 resolution, it obtains a 384-dimension feature embedding. And for the multi-stage NesT, which is used in the whole Sec. 5, consists of three stages and each stage contains two transformer encoder layers. Given input image with 80×80 resolution, it obtains a 512-dimensional feature embedding.

C More Training Implementation Details

C.1 Datasets Details

Following [5,6,39], we evaluate ViTs with SUN on three widely used few-shot benchmarks, i.e., CIFAR-FS [3], *mini*ImageNet [31] and *tiered*ImageNet [26] datasets.

***mini*ImageNet** [31] contains 100 different categories chosen from ImageNet-1k dataset [8], and each category includes 600 samples. Here we follow [5,6,39] and split *mini*ImageNet into 64/16/20 classes for *train*, *val* and *test* sets, respectively.

***tiered*ImageNet** [26] is a larger subset of ImageNet-1k dataset, which totally includes 779,165 images from 608 different categories. Specifically, this dataset includes 351 classes for *training* set, 97 classes for *validation* set and 160 classes for *test* set, respectively.

CIFAR-FS [3] is built upon CIFAR-100 dataset [13], which is divided into 64, 16 and 20 categories for training, validation and testing, respectively. Analogous to *mini*ImageNet, each category includes 600 different images.

C.2 Training Details

For meta-training phase, we use AdamW [21] with a learning rate of 5e-4 and a cosine learning rate scheduler [20] to train our meta learner f for 800, 800, 300 epochs on *mini*ImageNet, CIFAR-FS and *tiered*ImageNet, respectively. For augmentation, we use Spatial-Consistent Augmentation in Sec. 4. For location-specific supervision on each patch, we only keep the top-k (e.g., $k = 5$) highest confidence in $\hat{\mathbf{s}}_{ij}$ to reduce the label noise. To train the teacher f_g, we employ the same augmentation strategy in [29], including random crop, random augmentation, mixup, etc. Meanwhile we adopt AdamW [21] with the same parameters as above to train 300 epochs. For meta-tuning, we simply utilize the optimizer and the hyper-parameters used in Meta Baseline, e.g., SGD with learning rate of 1e-3 to finetune the meta-learner f for 40 epochs. Moreover, we use relatively large drop path rate 0.5 to avoid overfitting for all training. This greatly differs from conventional setting on drop path rate where it often uses 0.1. Following conventional supervised setting [34,11,36] , we also use a three-layer convolution block [9] with residual connection to compute patch embedding. This conventional stem has only $\sim$0.2M parameters and is much smaller than ViT backbone. All programs are implemented by PyTorch toolkit [25], and All experiments are conducted on two NVIDIA A100 GPUs.

C.3 Implementation Details of our Analysis in Sec. 3

[6]+CNN [32]. Here we aim to introduce inductive bias via explicitly introducing CNN layers. Generally, adding CNN layers has two methods: 1) directly inserting CNN layers after the transformer layers (like Visformer [7]); 2) adding independent CNN modules, and fuse the features from both transforme layer and CNN modules. Here we mainly discuss 2). Specifically, with given transformer layer T and a CNN module C (the input and output dimensions of T and C are same), given the input image feature $\mathbf{z}$, the updated feature is obtained by $\mathbf{z}' = T(\mathbf{z}) + C(\mathbf{z})$. We use this combination to replace each transformer layer in the original ViT, and obtain [6]+CNN.

[6]+DrLoc [18]. Here we introduce the implementation detail of [6]+DrLoc [18] mentioned in Sec. 5.3 of the main paper. With given meta-learner f, we additionally introduced a three-layer MLP projection head h and the output dimension

Table 2. Details of the Spatial-Consistent Augmentation (SCA) proposed in Sec. 4, where p means the random probability to perform the corresponding augmentation.

Spatial-Only Transformation	Non-Spatial Transformation
Random Crop and Resize	ColorJitter (brightness=0.4, contrast=0.4, saturation=0.4)
Random Horizontal Flip, p=0.5	Gaussian Blur, p=0.5
Random Rotation, p=0.2	Solarization, p=0.5
/	GrayScale, p=0.2
/	Random Erasing

of h is 2, which indicates the relative distance between two local tokens on x-axis and y-axis. For given local patches $[\mathbf{z}_{\text{cls}}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_K]$, we first sample m (e.g., $m = 64$) different token pairs $\{(\mathbf{z}_i, \mathbf{z}_j)\}$. For each $(\mathbf{z}_i, \mathbf{z}_j)$, we calculate the relative distance by $(\delta x', \delta y') = h(\text{concat}(\mathbf{z}_i, \mathbf{z}_j))$. Since the ground-truth of relative distance $(\delta x, \delta y)$ can be directly calculated, the overall training objective can be written as:

$$\mathcal{L}_{\text{drloc}} = H(g_{\text{global}}(\mathbf{z}_{\text{global}}), \mathbf{y}_i) + L_1(\delta x', \delta x) + L_1(\delta y', \delta y). \tag{1}$$

[6]+CNN Distill. Here we aim to introduce CNN-alike inductive bias from a pretrained CNN. Specifically, we first train a ResNet-12 feature extractor f_0^{CNN} with corresponding global classifier g_0^{CNN} on $\mathbb{D}_{\text{base}}$, and use them to teach ViT feature extractor f and corresponding global classifier g_{global} via knowledge distillation [10]. Following the description in Sec. 4.1, given f_0^{CNN} with g_0^{CNN}, as well as the target meta-learner f with global classifier g_{global}, we denote the the classification result from the global average pooling of all patch tokens calculated by g_0^{CNN} is $g_0^{\text{CNN}}(\mathbf{z}_{0,\text{global}})$. Analogously, that calculated by target classifier g_{global} is denoted as $g_{\text{global}}(\mathbf{z}_{\text{global}})$. Thus, the training objective is formed as:

$$\mathcal{L}_{\text{distill}} = H(g_{\text{global}}(\mathbf{z}_{\text{global}}), \mathbf{y}_i) + \text{JSD}(g_0^{\text{CNN}}(\mathbf{z}_{0,\text{global}}), g_{\text{global}}(\mathbf{z}_{\text{global}})), \tag{2}$$

where $\text{JSD}(\cdot, \cdot)$ indicates JS-Divergence between $g_0^{\text{CNN}}(\mathbf{z}_{0,\text{global}})$ and $g_{\text{global}}(\mathbf{z}_{\text{global}})$.

C.4 More Training Details of SUN

For meta-training phase, we use AdamW [21] with a learning rate of 5e-4 and a cosine learning rate scheduler [20] to train our meta learner f. Specifically, we train our model for 800, 800, 300 epochs on *mini*ImageNet, CIFAR-FS and *tiered*ImageNet, respectively. For data augmentation, we use Spatial-Consistent Augmentation in Sec. 4. To completely describe the spatial-consistent augmentation, Table 2 lists the detailed augmentation for both spatial-only part and non-spatial part. For location-specific supervision on each patch, we only keep

Table 3. Comparison with SoTA few-shot learning methods under 5-way few-shot classification setting, where SUN-F is our proposed SUN with FEAT [35] as meta-tuning phase. The results of the best 2 methods are in bold font.

Method	Classifier Params	*mini*ImageNet		*tiered*ImageNet		CIFAR-FS	
		1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
ResNet-12/18 as feature extractor							
MetaOptNet [14]	0	64.09±0.62	80.00±0.45	65.81±0.74	81.75±0.53	72.00±0.70	84.20±0.50
DeepEMD [37]	0	65.91±0.82	82.41±0.56	71.16±0.80	86.03±0.58	46.47±0.70	63.22±0.71
FEAT [35]	1.05M	66.78±0.20	82.05±0.14	70.80±0.23	84.79±0.16	-	-
TADAM [24]	1.23M	58.50±0.30	76.70±0.30	-	-	-	-
Rethink-Distill [28]	225K	64.82±0.60	82.14±0.43	71.52±0.69	86.03±0.49	73.90±0.80	86.90±0.50
DC [16]	224K	61.26±0.20	79.01±0.13	-	-	-	-
MTL [27]	0	61.20±1.80	75.50±0.80	-	-	-	-
CloserLook++ [5]	131K	51.87±0.77	75.68±0.63	-	-	-	-
Meta-Baseline [6]	0	63.17±0.23	79.26±0.17	68.62±0.27	83.29±0.18	-	-
Neg-Cosine [17]	131K	63.85±0.81	81.57±0.56	-	-	-	-
AFHN [15]	359K	62.38±0.72	78.16±0.56	-	-	68.32±0.93	81.45±0.87
Centroid [1]	10K	59.88±0.67	80.35±0.73	69.29±0.56	85.97±0.49	-	-
RE-Net [12]	430K	**67.60±0.44**	**82.58±0.30**	71.61±0.51	85.28±0.35	74.51±0.46	86.60±0.32
TPMN [33]	16M	**67.64±0.63**	**83.44±0.43**	**72.24±0.70**	**86.55±0.63**	**75.50±0.90**	**87.20±0.60**
NesT ViT as feature extractor							
CloserLook++ [5]	180K	49.23±0.43	66.57±0.39	59.13±0.46	77.88±0.39	63.89±0.49	80.43±0.37
Meta-Baseline [6]	0	54.57±0.46	69.85±0.38	63.73±0.47	79.33±0.38	68.05±0.48	81.53±0.36
BML [39]	180K	59.35±0.45	76.00±0.35	66.98±0.50	83.75±0.34	67.51±0.48	82.17±0.36
SUN-F (Ours)	1.05M	66.60±0.44	81.90±0.32	**72.66±0.51**	**87.08±0.33**	**77.87±0.46**	**88.75±0.33**

the top-k (e.g., $k = 5$) highest confidence in $\hat{\mathbf{s}}_{ij}$ to reduce the label noise. To train the teacher f_g (the combination of feature extractor f_0 with global classifier g_0), we employ the same augmentation strategy in [29], including random crop, random augmentation, mixup, etc. Meanwhile, we adopt AdamW [21] with the same settings as above to train it for 300 epochs. For meta-tuning, we simply utilize the same optimizer and settings as Meta-Baseline, i.e., SGD with learning rate of 1e-3 to finetune the meta-learner f for 40 epochs. Moreover, we use relatively large drop path rate 0.5 to avoid overfitting for all training. This greatly differs from conventional setting on drop path rate where it often uses 0.1 [29,30]. Following conventional supervised setting [34,11,36], we also use a three-layer convolution block [9] with residual connection to compute patch embedding. This conventional stem has only ∼0.2M parameters and is much smaller than ViT backbone.

D Using FEAT as Meta-Tuning of SUN

As mentioned in Sec. 4.2 of the main paper, we also investigate different meta-tuning methods, such as introducing the task-specific few-shot learning stage of FEAT [35] as our meta-tuning phase. With the same FEAT, our SUN framework still shows superiority over other few-shot learning frameworks. Here we first give a brief introduction of FEAT, and then compare the SUN with FEAT with existing methods to demonstrate the superiority our method.

Generally, the goals of FEAT are two-fold: 1) all feature embeddings from the support set $\mathbf{S}$ of each task τ should be aligned by a permutation invariant function to obtain more discriminative prototypes, and 2) aligned feature embedding of each query should be similar to embeddings with the same class and

dissimilar to those of other classes. Thus, it first introduces a self-attention layer A to obtain aligned classification prototypes $\mathbf{w}'_k = A(\{\mathbf{w}_k, \forall 1 \leq k \leq c\})$, and then uses Eqn. (3) to calculate the confidence score $\mathbf{p}'_k$ for each query $\mathbf{x}$,

$$\mathbf{p}'_k = \frac{\exp(\gamma \cdot \cos(GAP(f(\mathbf{x})), \mathbf{w}'_k))}{\sum_{k'} \exp(\gamma \cdot \cos(GAP(f(\mathbf{x})), \mathbf{w}'_k))}, \tag{3}$$

and then obtains the classification prediction $\mathbf{p}^{\mathrm{F}}_{\mathrm{x}} = [\mathbf{p}'_1, \cdots, \mathbf{p}'_c]$. Meanwhile, for each class $k \in c$, FEAT introduces contrastive learning loss among query embeddings for each task τ. Specifically, FEAT calculates the class center $\mathbf{q}_c$ as $\mathbf{q}_c = \sum A(\{f(\mathbf{x}'), \mathbf{x}' \in \tau_k\})/(N_q + N_k)$, where τ_c indicates all query and support images of class c, and A is the same self-attention layer mentioned above. Thus for each query $\mathbf{x}$, FEAT also enforces $f(x)$ aligned by A to be close to the corresponding class center $\mathbf{q}_c$, then we obtain:

$$\mathbf{p}^{\mathrm{aux}}_{\mathbf{x}} = \left\{ \frac{\exp(\gamma \cdot \langle f(\mathbf{x}) \cdot \mathbf{q}_c) \rangle}{\sum_{c'} \exp(\gamma \cdot \langle f(\mathbf{x}) \cdot \mathbf{q}_{c'} \rangle)}, \forall c \in \mathbf{S} \right\}. \tag{4}$$

Finally, it minimizes the classification loss $\mathcal{L}_{\text{few-shot}} = H(\mathbf{p}^{\mathrm{F}}_{\mathrm{x}}, \mathrm{y}_{\mathbf{x}}) + H(\mathbf{p}^{\mathrm{aux}}_{\mathbf{x}}, \mathrm{y}_{\mathbf{x}})$, where $\mathrm{y}_{\mathbf{x}}$ is the classification label of $\mathbf{x}$ w.r.t. $\mathbf{S}$. By using this meta-tuning method, we term our method "SUN-F".

Analogous to "SUN-M" stated in Sec. 5.4, we evaluate SUN-F using NesT [38] on three diferent datasets, i.e., *mini*ImageNet [31], *tiered*ImageNet [26] and CIFAR-FS [3]. The detailed evaluation results are given in Table 3. With the

Table 4. Comparison results among DeepEMD [37], COSOC [22] and SUN under 5-way few-shot classification setting on *mini*ImageNet.

Methods	DeepEMD [37]	COSOC [22]	SUN-D (Ours)
5-way 1-shot	68.77±0.29	69.28±0.49	**69.56±0.44**
5-way 5-shot	84.13±0.53	85.16±0.42	**85.38±0.49**

same NesT as meta-learner f, our SUN-F achieves the best 5-way 1-shot an 5-way 5-shot accuracy on the three datasets. Specifically, SUN-F outperforms BML by 7.6%, 5.7%, 10.3% in terms of 5-way 1-shot accuracy on *mini*ImageNet, *tiered*ImageNet, CIFAR-FS *test* sets, respectively. Moreover, SUN-F also performs very competitively in comparison with state-of-the-art CNN-based few-shot learning methods. Specifically, on *tiered*ImageNet under 5-way 1-shot and 5-shot settings, our SUN-F respectively obtains 72.66% and 87.08%, and respectively improves ∼0.4% and ∼0.5% over the SoTA TPMN [33]. On CIFAR-FS dataset, our SUN-F obtains 77.87% and 88.75% in terms of 1-shot accuracy and 5-shot accuracy, which significantly outperforms all the state-of-the-art methods by at least 2.3% in terms of 1-shot accuracy. Meanwhile, our SUN-F also obtains 66.60% 1-shot accuracy on *mini*ImageNet *test* set, which also surpasses most of CNN-based few-shot learning methods.

Table 5. Effect of drop path rate in teacher ViT model on *mini*ImageNet.

drop path rate p_{dpr}	0.1	0.2	0.3	**0.5**	0.8
5-way 1-shot	60.10±0.45	60.47±0.45	62.02±0.45	**62.40±0.44**	61.73±0.44
5-way 5-shot	77.44±0.33	77.97±0.34	79.29±0.31	**79.45±0.32**	78.12±0.32

E Using DeepEMD as Meta-Tuning of SUN

Besides, ViT with SUN inherently supports incorporating with dense prediction methods to conduct meta-tuning phase. There are two ways to leverage dense feature: **a)** selecting the foreground patch tokens with the local patch scores $g_{\text{local}}(\mathbf{z})$, then calculating the global token (via global average pooling) for classification; and **b)** simply replacing the meta-tuning method with dense prediction methods like DC [16] or DeepEMD [37]. Here we mainly discuss b) and use DeepEMD [37] as an example. From Table 4, SUN-D outperfroms DeepEMD by ∼1% on both 5-way 1-shot and 5-shot accuracy. The results indicate that ViTs with SUN can also leverage dense features to conduct few-shot classification and perform better than CNN counterpart. Moreover, the derived SUN-D achieves 69.56% 5-way 1-shot accuracy which is slightly higher than the state-of-the-art COSOC [22]. Note that DeepEMD is not applicable to COSOC, since DeepEMD uses dense feature while COSOC uses global tokens; while SUN can incorporate with various methods. Thus we believe that SUN can be incorporated with COSOC and achieves better performance than SUN and COSOC.

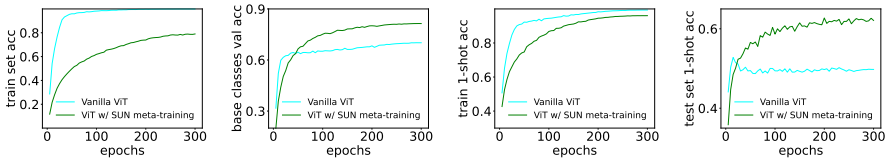
F More Ablation Study

F.1 More Analysis of Meta-Training Phase

To further analyze our meta-training phase, we plot the accuracy curves of the meta-training method in [5] and our meta-training phase in Fig. 1(a)∼1(d). As shown in Fig. 1(c) and 1(d), ViT with SUN achieves higher classification accuracy on novel classes. Besides, as shown in Fig. 1(b), SUN also obtains ∼11% improvement on base classes than meta-training method [5]. This observation inspires us to rethink the training paradigm of ViTs for few-shot classification, such that ViTs can generalize well on both base and novel categories.

F.2 Drop Path Rate Analysis

As mentioned in Sec. C.2, we also show the effect of drop path rate p_{dpr}. Here we use the teacher ViT model f_0 in meta-training phase to investigate. Table 5 shows that a relatively large drop path rate (e.g., $p_{dpr} = 0.5$) gives highest accuracy, since it can well mitigate over-fitting and is suitable for few-shot learning problems where training samples are limited. Based on Table 5 and the observation in Sec. 3, a possible explanation of choosing $p_{dpr} = 0.5$ is that the ViTs



(a) base *train* acc (b) base *val* acc (c) *train* 1-shot acc (d) *test* 1-shot acc

Fig. 1. Accuracy of ViTs (w/ or w/o SUN meta-training phase) on *mini*ImageNet. ViTs with SUN meta-training generalize better on both base and novel categories.

Table 6. Ablation study of training epochs of meta-training phase on *mini*ImageNet and *tiered*ImageNet, where "epochs" means training epochs in the meta-training phase.

Method	epochs	***mini*ImageNet**		***tiered*ImageNet**	
		5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
SUN Meta-Training	300	63.66±0.45	80.14±0.32	72.26±0.49	86.47±0.34
SUN Meta-Training	800	**64.84±0.45**	**80.96±0.32**	**72.34±0.49**	**86.57±0.34**

with small p_{dpr} may overfit to base classes during meta-training while ViTs with too large p_{dpr} may underfit to training samples.

F.3 Effect of Patch Embedding

Now we analyze the effect of overlapped patch embedding in Sec. C.2. For comparison, we replace the overlapped patch embedding in ViT by the vanilla non-overlapped patch embedding, and use the same meta-training method. As shown in Table 6 in the main paper, compared to type (a) using overlapped embedding, ViT w/o overlapped embedding achieves 59.70% and 77.19% in terms of 1-shot and 5-shot accuracy respectively. This result shows that overlapped patch embedding benefits the generalization ability on novel categories.

F.4 More Training Epochs in Meta-Training

Moreover, we also investigate the effect of training epochs during meta-training phase. The motivation comes from two aspects: 1) during meta-training, vanilla ViT faces severe generalization problems after 300 epochs training while ResNet-12 is opposite (see Fig. 2 in the main paper for details), 2) after fixing the generalization issue, ViTs with large-scale training tend to benefit from more training epochs. Thus we aim to investigate whether ViT with SUN benefits from more training epochs.

Following the widely used 300/800 training epochs in previous ViT works [4,2], we evaluate our SUN meta-training with 300/800 epochs on *mini*ImageNet (small size) and *tiered*ImageNet (large size). And during testing, we follow [6]

Table 7. Comparison between SUN meta-training phase without global JS-Divergence constraint and SUN meta-training phase with global JS-Divergence constraint on *mini*ImageNet, where "JSD" means further adding JS-Divergence between $g_0(\mathbf{z}_{0,\text{global}})$ and $g_{\text{global}}(\mathbf{z}_{\text{global}})$ in the meta-training phase.

Method	JSD	***mini*ImageNet**	
		5-way 1-shot	5-way 5-shot
SUN Meta-Training		**64.84±0.45**	**80.96±0.32**
SUN Meta-Training	✓	64.56±0.44 (-0.28%)	80.76±0.31 (-0.20%)

to evaluate the 5-way 1-shot and 5-way 5-shot accuracy on corresponding meta-learner f. Evaluation results are given in Table 6. For *mini*ImageNet, after introducing more training epochs, ViT with SUN outperforms that with 300 epochs by 1.2% and 0.8% in terms of 5-way 1-shot and 5-way 5-shot accuracy. This observation indicates that more training epochs can improve the generalization ability on novel categories of ViT with SUN. And for the relative larger *tiered*ImageNet, introducing more epochs only leads to 0.1% accuracy improvement. A possible explanation is that larger dataset may inherently benefit to the generalization ability, thus the contribution from more training epochs is limited. Therefore, for *mini*ImageNet and CIFAR-FS, we use 800 epochs for meta-training phase; and for *tiered*ImageNet, to trade off the training time as well as the classification accuracy, we keep using 300 epoch during meta-training.

F.5 Is Adding JS-Divergence (JSD) Benefit to SUN?

Denote by f_g the given teacher model consisting of a feature extractor f_0 and a classifier g_0, and f the target meta-learner with global classifier g_{global}. Following the description in Sec. 4.1, we denote the the classification result from the global average pooling of all patch tokens calculated by g_0 is $g_0(\mathbf{z}_{0,\text{global}})$. Analogously, that calculated by target classifier g_{global} is denoted by $g_{\text{global}}(\mathbf{z}_{\text{global}})$. Previous knowledge distillation works [10,28] mainly focus on minimizing the JS-Divergence between $g_0(\mathbf{z}_{0,\text{global}})$ and $g_{\text{global}}(\mathbf{z}_{\text{global}})$ and achieve better classification performance. Therefore, we focus on the meta-training phase of our SUN and conduct the ablation study of JS-Divergence on *mini*ImageNet to evaluate whether it is essential for SUN or not. Specifically, during meta-training without JSD, we keep using $\mathcal{L}_{\text{SUN}}$ as training loss; while during meta-training with JSD, we use $\mathcal{L}_{\text{SUN+JSD}} = \mathcal{L}_{\text{SUN}} + \text{JSD}(g_0(\mathbf{z}_{0,\text{global}}), g_{\text{global}}(\mathbf{z}_{\text{global}}))$ as training loss. And during testing, we follow [6] to evaluate the 5-way 1-shot and 5-way 5-shot accuracy on corresponding meta-learner f. The evaluation results are shown in Table 7. After introducing the global JS-Divergence constraint, meta-training with JSD drops 0.3% and 0.2% in terms of 5-way 1-shot and 5-way 5-shot accuracy. A possible explanation is that adding JS-Divergence may enforce the $g_{\text{global}}(\mathbf{z}_{\text{global}})$ to be similar to $g_0(\mathbf{z}_{0,\text{global}})$, thus somewhat ignores some knowledge from the location-specific supervision, and then slightly impairs the

generalization ability on novel classes. Thus, in the final SUN framework, the global JS-Divergence constraint is not included.

G t-SNE Visualization Results

Additionally, to qualitatively analyze the effects of our SUN, we also illustrate the t-SNE [23] results of ViT with Meta-Baseline [6] (short for "Baseline" in Fig. 2) and ViT with our SUN (short for "**SUN**" in Fig. 2). Following Sec. 5.2 and Sec. 5.5 in the main paper, we use NesT [38] as our ViT feature extractor.

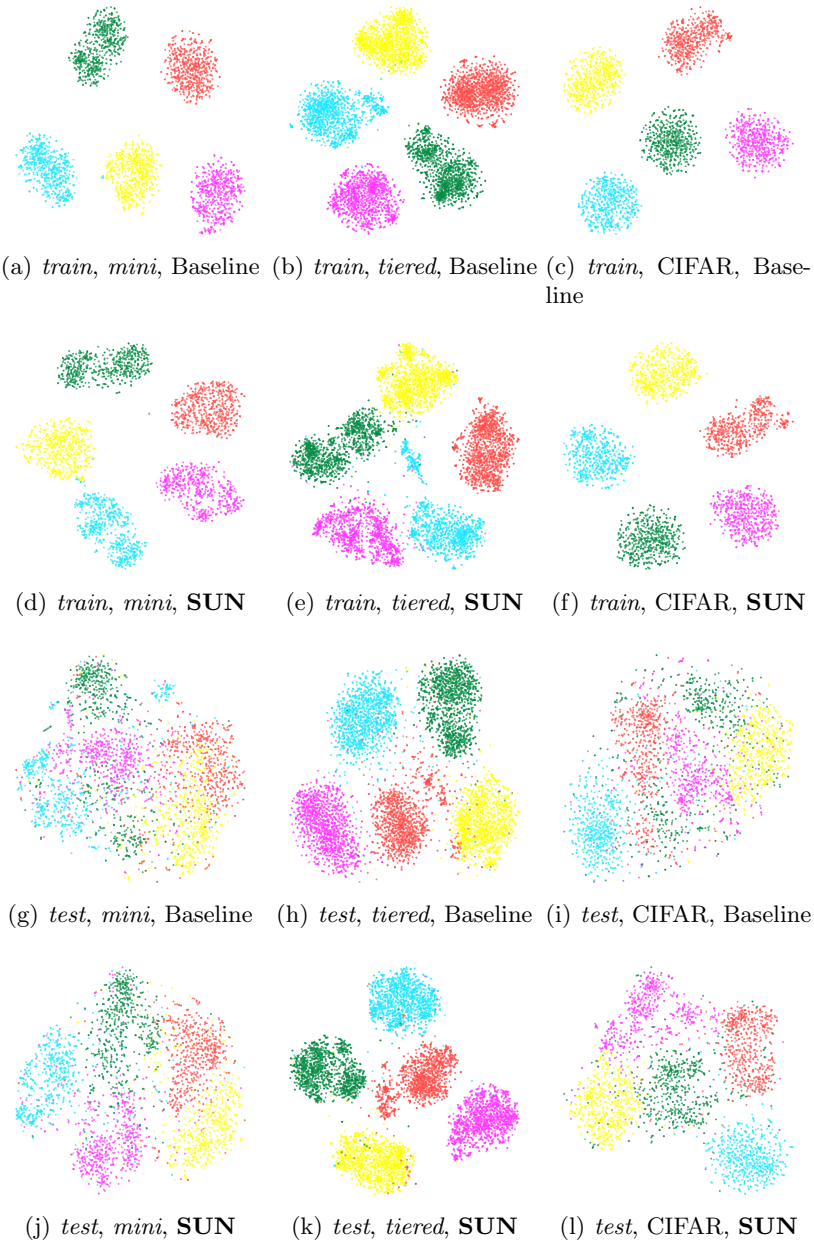
Detailed visualization results are shown in Fig. 2. Specifically, for each comparison pair (e.g., Fig. 2(g) and Fig. 2(j)), we keep using the same 5 categories for visualization. And for each category, we select the same 300 samples for visualization. According to these t-SNE visualization results, our ViT with SUN achieves the similar embedding grouping ability on base classes from the training set, but obtains better grouping ability for novel classes on test set. Especially, as shown in Fig. 2(g) and Fig. 2(j), the embeddings from novel classes extracted by "baseline" tend to be mixed together, but those from ViT with "SUN" can be separated into approximately 5 different groups. These observations also demonstrate the generalization ability of ViT with SUN on both base and novel categories, further verifying the effectiveness of SUN.

References

1. Afrasiyabi, A., Lalonde, J.F., Gagné, C.: Associative alignment for few-shot image classification. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) Computer Vision – ECCV 2020. pp. 18–35. Springer International Publishing, Cham (2020)
2. Bao, H., Dong, L., Piao, S., Wei, F.: BEit: BERT pre-training of image transformers. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=p-BhZSz59o4
3. Bertinetto, L., Henriques, J.F., Torr, P., Vedaldi, A.: Meta-learning with differentiable closed-form solvers. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=HyxnZh0ct7
4. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the International Conference on Computer Vision (ICCV) (2021)
5. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C., Huang, J.B.: A closer look at few-shot classification. In: International Conference on Learning Representations (2019)
6. Chen, Y., Liu, Z., Xu, H., Darrell, T., Wang, X.: Meta-baseline: Exploring simple meta-learning for few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9062–9071 (October 2021)
7. Chen, Z., Xie, L., Niu, J., Liu, X., Wei, L., Tian, Q.: Visformer: The vision-friendly transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 589–598 (October 2021)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)

9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770–778 (2016)
10. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. In: NIPS Deep Learning and Representation Learning Workshop (2015), http://arxiv.org/abs/1503.02531
11. Jiang, Z., Hou, Q., Yuan, L., Daquan, Z., Shi, Y., Jin, X., Wang, A., Feng, J.: All tokens matter: Token labeling for training better vision transformers. In: Advances in Neural Information Processing Systems (2021)
12. Kang, D., Kwon, H., Min, J., Cho, M.: Relational embedding for few-shot classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8822–8833 (October 2021)
13. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
14. Lee, K., Maji, S., Ravichandran, A., Soatto, S.: Meta-learning with differentiable convex optimization. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 10649–10657 (2019)
15. Li, K., Zhang, Y., Li, K., Fu, Y.: Adversarial feature hallucination networks for few-shot learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
16. Lifchitz, Y., Avrithis, Y., Picard, S., Bursuc, A.: Dense classification and implanting for few-shot learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
17. Liu, B., Cao, Y., Lin, Y., Li, Q., Zhang, Z., Long, M., Hu, H.: Negative margin matters: Understanding margin in few-shot classification. In: European Conference on Computer Vision. pp. 438–455 (2020)
18. Liu, Y., Sangineto, E., Bi, W., Sebe, N., Lepri, B., Nadai, M.D.: Efficient training of visual transformers with small datasets. In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems (2021), https://openreview.net/forum?id=SCN8UaetXx
19. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 10012–10022 (October 2021)
20. Loshchilov, I., Hutter, F.: Sgdr: Stochastic gradient descent with warm restarts. International Conference on Learning Representations (2017)
21. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (2019)
22. Luo, X., Wei, L., Wen, L., Yang, J., Xie, L., Xu, Z., Tian, Q.: Rectifying the shortcut learning of background for few-shot learning. In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems (2021), https://openreview.net/forum?id=N1i6BJzouX4
23. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of Machine Learning Research **9**(86), 2579–2605 (2008), http://jmlr.org/papers/v9/vandermaaten08a.html
24. Oreshkin, B., Rodríguez López, P., Lacoste, A.: Tadam: Task dependent adaptive metric for improved few-shot learning. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper/2018/file/66808e327dc79d135ba18e051673d906-Paper.pdf

25. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
26. Ren, M., Ravi, S., Triantafillou, E., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=HJcSzz-CZ
27. Sun, Q., Liu, Y., Chua, T., Schiele, B.: Meta-transfer learning for few-shot learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019)
28. Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J.B., Isola, P.: Rethinking few-shot image classification: a good embedding is all you need? ECCV (2020)
29. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers and distillation through attention. In: International Conference on Machine Learning. vol. 139, pp. 10347–10357 (July 2021)
30. Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., Jégou, H.: Going deeper with image transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 32–42 (October 2021)
31. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Advances in Neural Information Processing Systems. pp. 3630–3638 (2016)
32. Weng, Z., Yang, X., Li, A., Wu, Z., Jiang, Y.G.: Semi-supervised vision transformers (2021)
33. Wu, J., Zhang, T., Zhang, Y., Wu, F.: Task-aware part mining network for few-shot learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8433–8442 (October 2021)
34. Xiao, T., Dollar, P., Singh, M., Mintun, E., Darrell, T., Girshick, R.: Early convolutions help transformers see better. In: Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (eds.) Advances in Neural Information Processing Systems (2021)
35. Ye, H.J., Hu, H., Zhan, D.C., Sha, F.: Few-shot learning via embedding adaptation with set-to-set functions. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8808–8817 (2020)
36. Yu, P., Chen, Y., Jin, Y., Liu, Z.: Improving vision transformers for incremental learning (2021)
37. Zhang, C., Cai, Y., Lin, G., Shen, C.: Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
38. Zhang, Z., Zhang, H., Zhao, L., Chen, T., , Arik, S.O., Pfister, T.: Nested hierarchical transformer: Towards accurate, data-efficient and interpretable visual understanding. In: AAAI Conference on Artificial Intelligence (AAAI) (2022)
39. Zhou, Z., Qiu, X., Xie, J., Wu, J., Zhang, C.: Binocular mutual learning for improving few-shot classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 8402–8411 (October 2021)



(a) *train*, *mini*, Baseline (b) *train*, *tiered*, Baseline (c) *train*, CIFAR, Baseline

(d) *train*, *mini*, **SUN** (e) *train*, *tiered*, **SUN** (f) *train*, CIFAR, **SUN**

(g) *test*, *mini*, Baseline (h) *test*, *tiered*, Baseline (i) *test*, CIFAR, Baseline

(j) *test*, *mini*, **SUN** (k) *test*, *tiered*, **SUN** (l) *test*, CIFAR, **SUN**

Fig. 2. t-SNE visualization results of ViT without SUN (i.e. Baseline) and ViT with SUN (i.e. **SUN**) on three different datasets, where fig (a)∼(f) demonstrate the results of base classes from *train* set and fig (g)∼(l) demonstrate those of novel classes from *test* set. ViT with SUN performs better on all datasets.