Sparse Spatial Transformers for Few-Shot Learning

Haoxing Chen, Huaxiong Li, Yaohui Li, Chunlin Chen
Nanjing University, Nanjing, China
{haoxingchen, yaohuili}@smail.nju.edu.cn, {huaxiongli, clchen}@nju.edu.cn

Abstract

Learning from limited data is a challenging task since the scarcity of data leads to a poor generalization of the trained model. The classical global pooled representation is likely to lose useful local information. Recently, many few shot learning methods address this challenge by using deep descriptors and learning a pixel-level metric. However, using deep descriptors as feature representations may lose the contextual information of the image. And most of these methods deal with each class in the support set independently, which cannot sufficiently utilize discriminative information and task-specific embeddings. In this paper, we propose a novel Transformer based neural network architecture called Sparse Spatial Transformers (SSFormers), which can find task-relevant features and suppress task-irrelevant features. Specifically, we first divide each input image into several image patches of different sizes to obtain dense local features. These features retain contextual information while expressing local information. Then, a sparse spatial transformer layer is proposed to find spatial correspondence between the query image and the entire support set to select task-relevant image patches and suppress task-irrelevant image patches. Finally, we propose to use an image patch matching module for calculating the distance between dense local representations, thus to determine which category the query image belongs to in the support set. Extensive experiments on popular few-shot learning benchmarks show that our method achieves the state-of-the-art performance.

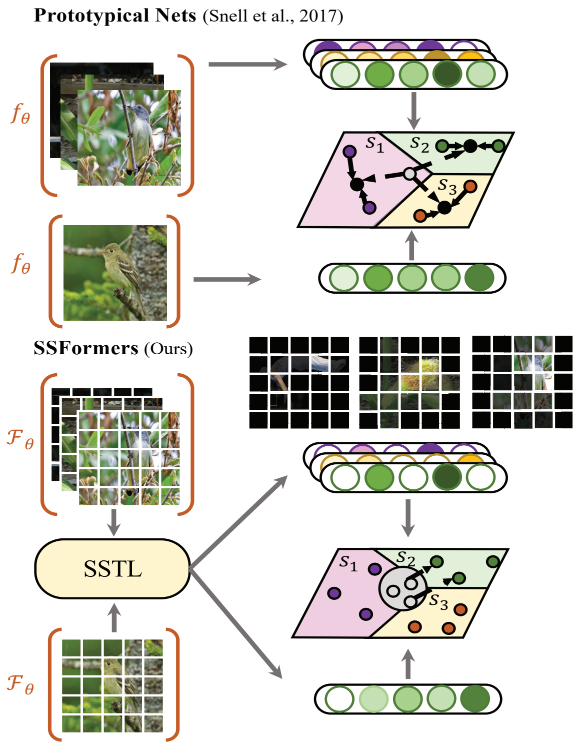

Figure 1. Prototypical Nets [27] learns a global-level representation in an appropriate feature space and uses Euclidean distance to measure similarities. On the contrary, our model first generates dense local representations through image patches, and then uses Sparse Spatial Transformer Layer (SSTL) to select task-relevant patches, generating task-specific prototypes. Finally, similarities are obtained by matching between attentioned image patches.

1. Introduction

With the availability of large-scale labeled data, visual understanding technology has made great progress in many tasks [2, 19, 34]. However, collecting and labeling such a large amount of data is time-consuming and laborious. Few-shot learning is committed to solving this problem, which enables deep models to have better generalization ability even on a small number of samples.

Recently, many few-shot learning methods have been proposed, which can be roughly divided into two categories: meta-learning [6, 9, 23] and metric-learning [3, 4, 17, 27, 28, 31]. The goal of meta-learning is to learn how to deal with new tasks in the process of learning multiple tasks [9]. Metric learning focuses on learning a good feature representation or relation measure [4, 27].

For feature representations, most of the existing metric-learning based methods [16,27,31] adopt global features for recognition, which may cause useful local information to be lost and overwhelmed. Recently, DN4 [17], MATANet [3] and DeepEMD [33] adopt dense feature representations (i.e., deep descriptors) for few-shot learning tasks, which have been verified to be more expressive and effective than using global features. Another branch that enhances image representation uses the attention mechanism to align the query image with the support set. For example, Cross Attention Network (CAN) [12] and SAML [11] use the semantic correlation between the support set and query image to highlight the target object.

For the relation measure, existing dense feature based methods usually adopt a pixel-level metric and the query image is taken as a set of deep descriptors. For example, in DN4 [17], for each query deep descriptor, they find its nearest neighbor descriptors in each support class. Also, CovaMNet [18] calculates a local similarity between each query deep descriptor and a support class by a covariance metric.

However, most existing methods use global features or deep descriptors, and they are not effective for few-shot image classification. Due to global features lose local information, and deep descriptors lose the contextual information of images. Moreover, the above methods all process each support class independently, and cannot use the context information of the entire task to generate task-specific features.

In this paper, we propose a novel transformer-based architecture for few-shot learning, called sparse spatial transformer (SSFormers), which extracts the spatial correlation between the query image and the current task (the entire support set), aiming to align task-relevant image patches and suppress task-irrelevant image patches. As shown in Figure 1, we first divide each input image into several patches and get dense local features. Second, we select task-relevant query patches by a two-way selection function, i.e., mutual nearest neighbour [10, 20]. And use selected query patches to align support classes. Finally, a patch matching module is proposed to measure the similarity between query images and aligned support classes. For each patch from a query image, the patch matching module calculates its similarity scores to the nearest neighbor patch in each aligned class prototype. Then, similarity scores from all query patches are accumulated as a patch-to-class similarity.

The main contributions of this work are summarized as follows:

1) We propose a novel *sparse spatial transformers* for few-shot learning, which can select task-relevant patches and generate a task-specific prototype.

2) We propose a *patch matching module* to get similarity between query images and task-specific prototypes. Experiments prove that it is more suitable for image patch-based feature representation than directly using the cosine similarity.

3) We conduct extensive experiments on popular few-shot learning benchmarks and show that the proposed model achieving competitive results compared to other state-of-the-art methods.

2. Related Work

Global Feature based Methods. The traditional metric-learning based few-shot learning methods use an additional global average pooling (GAP) layer to obtain the global feature representation at the end of the backbone and utilize different metrics for classification. MatchingNet [31] utilizes the cosine distance to measure the similarity between the query image and each support class. ProtoNet [27] takes the empirical mean as the prototype representation of each category and uses Euclidean distance as the distance metric. RelationNet [28] proposed a non-linear learnable distance metric. These methods based on global features will lose a lot of useful local information, which is harmful to classification tasks under few-shot learning settings.

Dense Feature based Methods. Another branch of metric-learning based methods uses pixel-level deep descriptors as feature representations. DN4 [17] uses the k-nearest neighbor algorithm to obtain the pixel-level similarity between images. MATANet [3] proposes a multi-scale task adaptive network to select task-relevant deep descriptors at multiple scales. DeepEMD [33] proposes a differentiable earth mover's distance to calculate the similarity between image patches. Our SSFormers also belong to this method based on dense features. A major difference in our method is that we divide input images into several patches of different sizes and extract features. Compared with global features, the features extracted by our method can express local information. And compared with deep descriptors, the extracted features contain context information.

Attention based Methods. CAN [12] proposes a cross-attention algorithm to highlight the common objects in the image pair. SAML [11] proposes a collect and select strategy to align the main objects in the image pair. RENet [13] improves network generalization performance over unseen categories from a relational perspective. SSFormers also be treated among the family of transformered-based methods as they also aligned query images and support sets. Differently, our SSFormers select task-relevant patches in the query image to align support set to query image by a sparse spatial cross attention algorithm.

Transformers based Methods. FEAT [32] first introduced Transformer [30] to few-shot learning. FEAT utilized Transformer to conduct support set sample relationships and generate task-specific support features. CrossTransformers [8] proposes to use a self-supervised learning algorithm to enhance the feature representation ability of the pre-trained backbone, and use a transformer to achieve alignment. Unlike them, SSFormers models the relationship between query images and support classes on a spatial scale, and introduces MNN [10] to select task-relevant patches. SSFormers are more efficient and explanatory.

3. Preliminary

We first introduce the problem definition of few-shot learning. Few-shot learning is dedicated to learning transferable knowledge between tasks and using the learned knowledge to solve new tasks. In the few-shot learning scenario, the task is usually set in the form of *N*-way *M*-shot, where *N* is the number of categories and *M* is the number of labeled samples in each category. Under this setting, the model is trained on a training set $\mathcal{D}_{train}$ with a large amount of labeled data. To learn transferable knowledge, we use episodic training mechanisms to train our model. The episodic training mechanism samples batched tasks from $\mathcal{D}_{train}$ for training. In each episode, we first construct query set $\mathcal{D}_Q = \{(x_i^q, y_i^q)\}_{i=1}^{N\times B}$ and support set $\mathcal{D}_S = \{(x_i^s, y_i^s)\}_{i=1}^{N\times M}$, where B is a hyperparameter that we need to fix in our experiments. Typically, B is set to 15 [12, 32]. Then our model predicts which support set category each sample in the query set belongs to. When the model training is complete, we sample tasks from unlabeled test sets $\mathcal{D}_{test}$ to verify the performance of the model.

4. Our Method

In this section, we first introduce our method for generating dense local representations. Then we describe our sparse spatial transformers layer, which spatially aligns query images and support classes. Finally, we describe the patch matching module (PMM), which is used to calculate the final similarities. The overview of our framework is shown in Figure 2.

4.1. Dense Local Feature Extractor

The metric-learning based few-shot learning method aims to find an effective feature representation and a good distance metric to calculate the similarity between images. Different from the methods that use global features, local representation based methods have achieved better results because the local representation contains richer and more transferable semantic information. The difference from previous work is that our SSFormers aims to establish hierarchical local representations for spatial comparison.

As illustrated in Figure 2, dense local representations extractor F_θ evenly divides the image into $H \times W$ patches, and each image patch is individually encoded by the backbone network to generate a feature vector. The feature vectors generated by all patches constitute the dense local representations set of each image. To generate hierarchical local representations, we adopt a pyramid structure in the experiments. Thus the feature representation of an input image x can be denoted as $F_\theta(x) \in \mathbb{R}^{K\times C}$, where C is the number of channels, and K is the number of all local patch representations. Specifically, we adopt two image patch division strategies of size 2×2 and 4×4 to obtain 20 dense local representations.

In each N-way M-shot few-shot image recognition task, for each support class, we have M samples and get *M* feature representations. Instead of using empirical mean of *M* feature representations [27] to obtain the class representation, we utilize all the patches in each support class, i.e., $S_n \in \mathbb{R}^{MK\times C}$, where S_n is the class representation of the n-th support class. The entire support set representation can be denoted as $S \in \mathbb{R}^{N\times MK\times C}$. Similarly, for a query image x_i^q, through F_θ, we can get feature representation $q = F_\theta(x_i^q) \in \mathbb{R}^{K\times C}$, where $i = \{1, ..., BN\}$.

4.2. Sparse Spatial Transformers Layer

Sparse spatial transformers aim to enhance the discriminant ability of local feature representations by modeling the interdependencies between different patches in the query image and the entire support set. In a *N*-way *M*-shot task, key k_S and value v_S are generated for support set feature S using two independent linear projection: the key projection head h_k: $\mathbb{R}^C \mapsto \mathbb{R}^{C'}$ and the value projection head h_v: $\mathbb{R}^C \mapsto \mathbb{R}^{C'}$. Similarly, the query image feature q is embedded using the value projection head h_v and another the query projection head h_q: $\mathbb{R}^C \mapsto \mathbb{R}^{C'}$ to obtain value v_q and query q_q.

Inspired by [10], which proposed the mutual nearest neighbor (MNN) algorithm to eliminate batch effects in single-cell RNA sequencing data. We argue that if the patch with the closest semantic distance to patch q_i is S_j, and the patch with the closest semantic distance to patch S_j is q_i, then they are likely to have similar local feature, where $q_i \in q_q, i \in \{1, ..., K\}$ and $S_j \in k_S, j \in \{1, ..., NMK\}$. On the other hand, if the closest patch of patch S_j is not q_i, then even if the closest descriptor of q_i is S_j, the actual relationship between them is relatively weak. In other words, the correlation between two patches is a function of mutual perception, not a function of one-way perception. Therefore, we can use this bidirectionality to select task-relevant patches in the current task.

We first calculated semantic relation matrix between query image and each support class n, and get $\mathbf{R}_n$:

$$\mathbf{R}_n = \frac{q_q \times k_{S_n}^\top}{\sqrt{C'}} \in \mathbb{R}^{K\times MK} \tag{1}$$

To find task-relevant patches, we concatenate all semantic relation matrixes $\mathbf{R}_n, n = \{1, ..., N\}$ to get $\mathbf{R} \in$

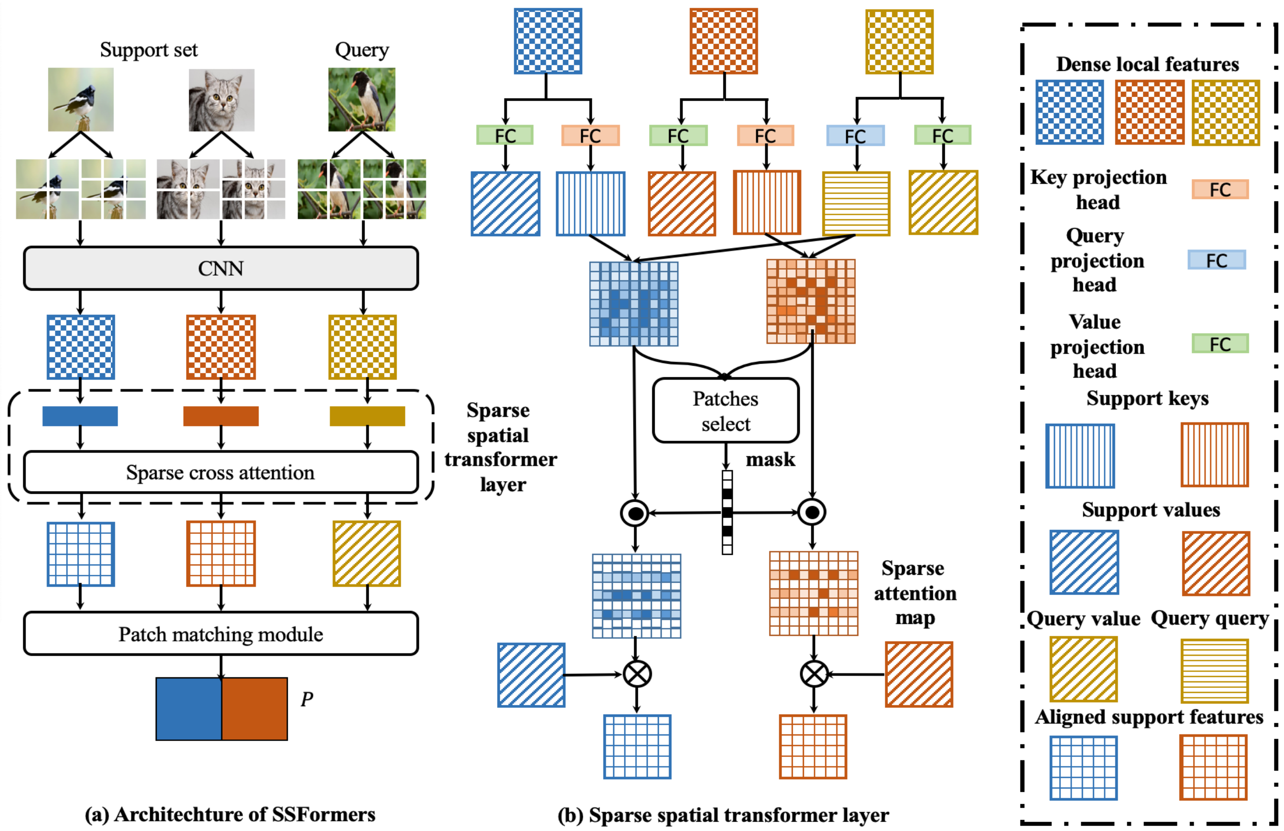

Figure 2. Illustration of the proposed SSFormers. We propose to generate dense local features and find task-relevant features by a sparse spatial transformers layer.

Algorithm 1 Training strategy of SSFormers

```
Require: Training set $\mathcal{D}_{train}$
 1: for all iteration=1, ..., MaxIteration do
 2:     Sample $N$-way $M$-shot task $(\mathcal{D}_Q, \mathcal{D}_S)$ from $\mathcal{D}_{train}$
 3:     Compute $S = F_\theta(\mathcal{D}_S) \in \mathbb{R}^{N \times MK \times C}$
 4:     for $i$ in $\{1, ..., NB\}$ do
 5:         Compute $q = F_\theta(x_i^q) \in \mathbb{R}^{K \times C}$
 6:         Compute sparse attention map by Eq. (1)-(5)
 7:         Obtain task-spacific prototype $v_{S_n|q}$ by Eq. (6)
 8:         Compute similarity $P_i$ by Eq. (7) and (8)
 9:     end for
10:     Obtain Cross Entropy loss $\mathcal{L} = \sum_{i=1}^{NB} \text{CE}(P_i, y_i^q)$
11:     Update parametres in SSFormers by SGD
12: end for
13: return Trained SSFormers
```

$\mathbb{R}^{K \times NMK}$. Each row in $\mathbf{R}$ represents the semantic similarity of each patch in the query image to all patches of all images in the support set.

Specifically, we propose a novel sparse spatial cross attention algorithm to find task-relevant patches in the query image. For each patch $q_i \in q_q$, we find its nearest neighbor n_q^i in k_S, and then find the nearest neighbor n_S^i of n_q^i in q_q. If $i = n_S^i$, then we consider q_i to be a task-relevant patch. After collecting all task-relevant patches in q_q, we can get the mask $m = [m_1; ...; m_K]$, which can be computed as:

$$n_q^i = \arg\max_j \mathbf{R}_{i,j} \tag{2}$$

$$n_S^i = \arg\max_k \mathbf{R}_{k,n_q^i} \tag{3}$$

$$m^i = \mathbb{1}(i = n_S^i) \tag{4}$$

where $\mathbb{1}$ is the indicator function: when $i = n_S^i$, $\mathbb{1}$ is equal to 1, otherwise it is 0. Using mask m and semantic relation matrix $\mathbf{R}_n$, we can get sparse attention map a^n and use it to align each support class n to query image q and get task-specific prototype $v_{S_n|q}$, which can be computed as:

$$a_n = m * \mathbf{R_n} \in \mathbb{R}^{K \times MK} \tag{5}$$

$$v_{S_n|q} = a_n \times v_{S_n}^\top \in \mathbb{R}^{K \times C'} \tag{6}$$

4.3. Patch Matching Module

Patching maching module is built as a similarity metric, which does not have any parameter to train. Given query value $v_q \in \mathbb{R}^{K \times C'}$ and the aligned prototype of class n $v_{S_n|q} \in \mathbb{R}^{K \times C'}$, we can get their patch-to-patch similarity matrix by:

$$\mathbf{D}^n = \frac{v_q \times v_{S_n|q}}{||v_q|| \cdot ||v_{S_n|q}||} \in \mathbb{R}^{K \times K} \tag{7}$$

Then, for each patch in v_q, we select the most similar patch in all patches from prototype $v_{S_n|q}$. We sum K selected patches as the similarity between the query image and support class n:

$$P^n = \sum_{i=1}^{K} \max_{j \in \{1,\ldots,K\}} \mathbf{D}^n_{i,j} \tag{8}$$

Under the N-way M-shot few-shot learning setting, we can get semantic similarity vectors $P \in \mathbb{R}^N$. The training procedure of SSFormers is shown in Algorithm 1.

5. Experiments

To evaluate the effectiveness of our method, we conduct extensive experiments on several common-used benchmarks for few-shot image recognition. In this section, we first present details about datasets and experimental settings in our network design. Then, we compare our method with the state-of-the-art methods on various few-shot learning tasks, i.e., standard few-shot learning, cross-domain few-shot learning. Finally, we conduct comprehensive ablation studies to validate each component in our network.

5.1. Datasets

We conduct few-shot image recognition problems on four popular benchmarks, i.e., *mini*ImageNet, *tiered*ImageNet, CIFAR-FS and FC100.

***mini*ImageNet** [31] is a subset randomly sampled from ImageNet and is an important benchmark in few-shot learning community. *mini*ImageNet consists of 60,000 images in 100 categories. We follow the standard partition settings [27], where there are 64/16/20 categories for training, validation and evaluation.

***tiered*ImageNet** [24] is also a subset random sampled from ImageNet, which consists of 779,165 images in 608 categories. All 608 categories are grouped into 34 broader categories. Following the same partition settings [12], we use 20/6/8 broader categories for training, validation, and evaluation respectively.

CIFAR-FS [1] is divided from CIFAR-100, which consists of 60,000 images in 100 categories. The CIFAR-FS is divided into 64, 16 and 20 for training, validation, and evaluation, respectively.

FC100 [22] is also divided from CIFAR-100, which is more difficult because it is more diverse. The FC100 uses a split similar to *tiered*ImageNet, where train, validation, and test splits contain 60, 20, and 20 classes.

5.2. Implementation Details

Backbone networks. For fair comparison, following [26], we employ *Conv-64F* and *ResNet12* as our model backbone. To generate pyramid dense features, we add a global average pooling layer at the end of the backbone, such that the backbone generates a vector for each input image patch. And we slightly expand the area of the local patches in the grid by 2 times to merge the context information, which is helpful to generate the local representations.

Training details. For *Conv-64F*, we train it from scratch. For *ResNet12*, the training process can be divided into two stages: pre-training and meta-training. Following [33], we apply a pre-train strategy. The backbone networks are trained on training categories with a softmax layer. In this stage, we apply data argumentation methods to increase the generalization ability of the model, i.e., color jitter, random crop, and random horizontal flip. The backbone network in our model is initialized with pre-trained weights, which are then fine-tuned along with other components in our model. In the meta-training stage, we conduct N-way M-shot tasks on all benchmarks, i.e., 5-way 1-shot and 5-way 5-shot. *Conv-64F* is optimized by Adam, and the initial learning rate is set to 0.1 and decay 0.1 every 10 epochs. And *ResNet12* is optimized by SGD, and the initial learning rate is set to 5e-4 and decay 0.5 every 10 epochs.

Evaluation. During the test stage, we random sample 10,000 tasks from the meta-testing set, and take averaged top-1 classification accuracy as the performance of methods.

5.3. Standard Few-shot Image Classification

To verify the effectiveness of our proposed SSFormers for few-shot image classification task, we conduct comprehensive experiments and compare our methods with other state-of-the-art methods. The experimental results are shown in Table 1 and Table 2. The results show that our method achieves the best results in almost all settings.

For ImageNet derivatives, compared with the classic attention-based method CAN [12], our model is around 4.4%/4.0% better than CAN on *mini*ImageNet with *ResNet12* for 1-shot and 5-shot tasks. Compared with the previous transformers-based method FEAT [32], which uses transformers to find task-specific features in the support set, our method achieves 1.4%/2.0% improvements on *tiered*ImageNet with *ResNet12*.

For CIFAR derivatives, our model also achieved competitive results. For example, our model is around 3.2%/3.6%, 16.6%/12.2% better than baseline Prototypical Nets [27] on CIFAR-FS and FC100 for 1-shot/5-shot tasks, respectively.

The reason why we can achieve this improvement is that SSFormers can find task-relevant patches in the current task and perform sparse spatial cross attention algorithms based on hierarchical dense representations.

Method	Venue	Backbone	*mini*ImageNet		*tiered*ImageNet	
			5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Prototypical Networks [27]	NeurIPS'17	Conv-64F	49.42±0.78	68.20±0.66	53.31±0.89	72.69±0.74
CovaMNet [18]	AAAI'19	Conv-64F	51.19±0.76	67.65±0.63	54.98±0.90	71.51±0.75
DN4 [17]	CVPR'19	Conv-64F	51.24±0.74	71.02±0.64	53.37±0.86	74.45±0.70
DSN [26]	CVPR'20	Conv-64F	51.78±0.96	68.99±0.69	53.22±0.66	71.06±0.55
DeepEMD† [33]	CVPR'20	Conv-64F	52.15±0.28	65.52±0.72	50.89±0.30	66.12±0.78
SSFormers	Ours	Conv-64F	55.00±0.22	70.55±0.17	55.54±0.19	73.72±0.21
Prototypical Networks [27]	NeurIPS'17	ResNet12	62.59±0.85	78.60±0.16	68.37±0.23	83.43±0.16
CAN [12]	NeurIPS'19	ResNet12	63.85±0.48	79.44±0.34	69.89±0.51	84.23±0.37
DSN [26]	CVPR'20	ResNet12	62.64±0.66	78.83±0.45	67.39±0.82	82.85±0.56
DeepEMD [33]	CVPR'20	ResNet12	65.91±0.82	82.41±0.56	71.16±0.87	83.95±0.58
FEAT [32]	CVPR'20	ResNet12	66.78±0.20	82.05±0.14	70.80±0.23	84.79±0.16
GLoFA [21]	AAAI'21	ResNet12	66.12±0.42	81.37±0.33	69.75±0.33	83.58±0.42
ArL [35]	CVPR'21	ResNet12	65.21±0.58	80.41±0.49	-	-
PSST [5]	CVPR'21	ResNet12	64.05±0.49	80.24±0.45	-	-
RENet [13]	ICCV'21	ResNet12	67.60±0.44	82.58±0.30	71.61±0.51	85.28±0.35
SSFormers	Ours	ResNet12	67.25±0.24	82.75±0.20	72.52±0.25	86.61±0.18

Table 1. Average classification accuracy of 5-way 1-shot and 5-way 5-shot tasks with 95% confidence intervals on *mini*ImageNet and *tiered*ImageNet. † denotes that it is our reimplementation under the same setting. (Top two performances are shown in red and blue.)

Model	Venue	Backbone	CIFAR-FS		FC100	
			5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
Prototypical Networks [27]	NeurIPS'17	Conv-64F	55.50±0.70	72.00±0.60	35.30±0.60	48.60±0.60
Relation Networks [28]	CVPR'18	Conv-256F	55.00±1.00	69.30±0.80	-	-
R2D2 [1]	ICLR'19	Conv-512F	65.30±0.20	79.40±0.10	-	-
Prototypical Networks [27]	NeurIPS'17	ResNet-12	72.20±0.70	83.50±0.50	37.50±0.60	52.50±0.60
TADAM [22]	NeurIPS'18	ResNet-12	-	-	40.10±0.40	56.10±0.40
MetaOptNet [15]	CVPR'19	ResNet-12	72.60±0.70	84.30±0.50	41.10±0.60	55.50±0.60
MABAS [14]	ECCV'20	ResNet-12	73.51±0.92	85.49±0.68	42.31±0.75	57.56±0.78
Fine-tuning [7]	ICLR'20	WRN-28-10	76.58±0.68	85.79±0.50	43.16±0.59	57.57±0.55
RENet [13]	ICCV'21	ResNet12	74.51±0.46	86.60±0.32	-	-
SSFormers	Ours	ResNet-12	74.50±0.21	86.61±0.23	43.72±0.21	58.92±0.18

Table 2. Experimental results compared with other methods on CIFAR-FS and FC100. (Top two performances are shown in red and blue.)

5.4. Semi-supervised Few-Shot Learning

We further verify the effectiveness of our model on more challenging semi-supervised few-shot learning tasks. Under semi-supervised few-shot learning settings, we can select image patches from unlabeled samples that meet the mutual perception function (Eq. (2)-(4)) with the current support set and add them to the support set to provide more support features. Specifically, the workflow of SSFormers-semi is as follows. For the support set n, we first search for all patches that satisfy the mutual perception function in unlabeled sets and put them into the set U_n.Then we use U_n to extend S_n: $S_n = \{S_n^1, ..., S_n^{MK}\} \bigcup U_n$. Then, we use the original SSFormers to calculate the similarity.

We use the same experiment setting in [25]. We use *Conv-64F* as our backbone and train SSFormers-semi on 300,000 tasks on *mini*ImageNet. The results are shown in Figure 3, where SSFormers-semi shows competitive results with classical baseline methods.

5.5. Ablation Study

Analysis of Our Method. Our model consists of different components: dense local feature extractor, sparse spatial transformer layer, and patch matching module. As shown in Table 3, we verify the indispensability of each component

Dense Local Feature	SSTL	PMM	Cosine Classifier	5-way 1-shot	5-way 5-shot
	✓	✓		63.15±0.20	79.15±0.25
✓		✓		66.84±0.47	79.72±0.50
✓	✓		✓	64.35±0.22	80.17±0.17
✓	✓	✓		67.25±0.24	82.75±0.20

Table 3. Ablation study on our model, we can find that each part of our model has important contribution. The experiments are conducted with *ResNet12* on *mini*ImageNet. (SSTL: Sparse Spatial Transformer Layer and PMM: Patch Matching Module)

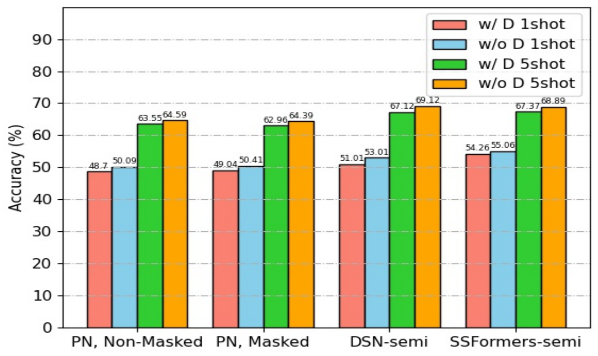

Figure 3. 5-way semi-supervised few-shot learning results on *mini*ImageNet. We show the results with (w/ D) and without *distractors*(w/o D). And we compare our methods with PN, Non-Masked [25], PN, Masked [25] and DSN-semi [26].

Embedding	5-way 1-shot	5-way 5-shot
5×5	65.20±0.24	80.07±0.26
4×4	66.37±0.23	81.06±0.25
3×3	65.17±0.22	81.69±0.20
2×2	65.18±0.22	80.10±0.16
$5 \times 5 + 3 \times 3$	66.38±0.23	81.82±0.24
$4 \times 4 + 3 \times 3$	66.08±0.24	81.50±0.26
$4 \times 4 + 2 \times 2$	67.25±0.24	82.75±0.20
$3 \times 3 + 2 \times 2$	67.05±0.22	82.53±0.18

Table 4. The 5-way, 1-shot and 5-shot classification accuracy (%) with different number of image patches on *mini*ImageNet.

on the *mini*ImageNet dataset. Note that when replacing the patch matching module, we calculate the cosine similarity spatially. The results show that every component in SS-Formers has a significant contribution. For example, without our sparse spatial transformer layer, the performance of the model will drop by 3.73% on 5-shot tasks.

Influence of the number of patches. During dividing input images into patches, we have to define the grid for patches. We select various grids and their combinations and conduct analysis experiments on *mini*ImageNet. As shown

Method	Self	Cross	1-shot	5-shot
ProtoNet			68.37±0.23	83.43±0.16
CAN		✓	69.89±0.51	84.23±0.37
FEAT		✓	70.80±0.23	84.79±0.16
RENet	✓	✓	71.61±0.51	85.28±0.35
SSFormers		✓	72.52±0.25	86.61±0.18

Table 5. The 5-way, 1-shot and 5-shot classification accuracy (%) with different attention methods on *tiered*ImageNet.

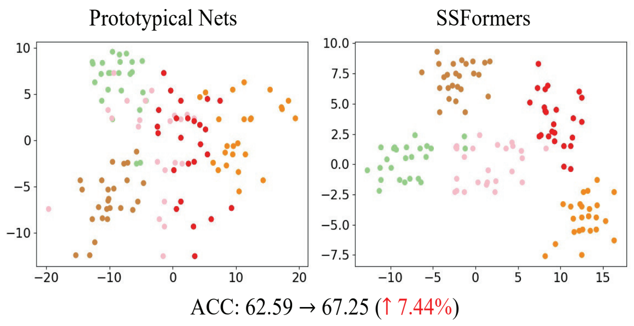

Figure 4. t-SNE visualization of features for 75 randomly sampled images from 5 randomly selected test classes of *mini*ImageNet dataset. In our case, the learned embeddings provide better discrimination for unseen test classes.

in Table 4, it is better to use a combination of grids of different sizes. A possible explanation is that the size of the main object in different images is different, and the use of a single size may lose context information, and make it difficult to generate high-level semantic representations.

Comparison with other attention methods. As shown in Table 5, our model has achieved state-of-the-art in all attention-based methods. Specifically, compared with RENet [13], our SSFormers only utilize cross attention, but still leads them in both 1-shot and 5-shot tasks.

Qualitative visualizations. We perform a t-SNE visualization of embeddings that generated by SSFormers from novel query images to demonstrate the effectiveness of our method (see Figure 4). We observe that our method main-

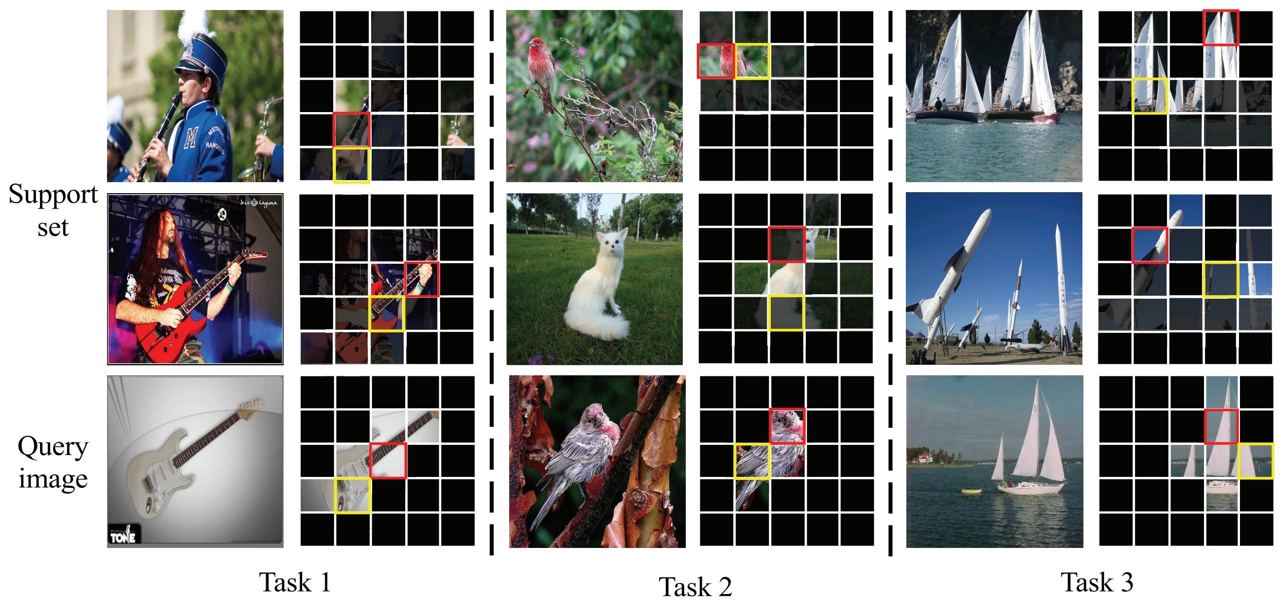

Figure 5. Visualization of the mask and sparese attention. Given 2-way 1-shot tasks, we plot the masked query image and attentioned support images (brighter colors mean higher weight). Within each query image, we choose two image patches (red and yellow boxes), and plot the image patch that best matches in each support class. This proves that our sparse spatial transformer layer can automatically highlight task-relevant areas.

	+ GaussianBlur ($\delta \in [0.1, 2]$)
Rethink-Distill	82.14→49.30
SSFormers	82.75→77.17
	+ PepperNoise ($r = 0.01$)
Rethink-Distill	82.14→63.97
SSFormers	82.75→66.74
	+ ColorJitter ($B = 0.8$)
Rethink-Distill	82.14→81.05
SSFormers	82.75→81.21

Table 6. Stability evaluation on *mini*ImageNet. Comparison with Rethink-Distill [29].

tains good class discrimination compared to prototype networks, even for unseen test classes. Moreover, the features generated by our method are more discriminative and the boundaries between categories are more obvious.

To further qualitatively evaluate the proposed SSFormers, we provide some visualization cases as shown in Figure 5. For each query image in the task, we plot the result of its mask, and it can be seen that through the mutual perception function (Eq. (2)-(4)), we can get task-relevant query image patches. For support sets, we plot the prototype generated after SSTL. It can be seen that our model can highlight task-relevant image patches and suppress task-irrelevant features.

Analyze of stability. A good model should have good robustness and be able to adapt to various environments. For this reason, we tested the stability of our model under three different attacks. As shown in Table 6, the performance of our SSFormers is relatively stable under various attacks.

Complexity analysis. The computational complexity of our SSFormers is $\mathcal{O}(NMCK^2)$, where N, M, C and K are the number of way, shot, the feature dimensionality, and the number of image patches, respectively. Compared to the complexity of CAN [12], i.e., $\mathcal{O}(NMCH^2W^2)$, where W and H are the size of feature maps, Our method has less computational complexity because $K^2 < H^2W^2$.

6. Conclusion

In this work, we argue that both global features and deep descriptors are not effective for few-shot learning since global features lose local information, and deep descriptors lose the contextual information of images. Moreover, a common embedding space fails to generate discriminative visual representations for a target task. We propose a novel sparse spatial transformers (SSFormers) for few-shot learning, which customizes task-specific prototypes via a transformer-based architecture. Experimental results demonstrate SSFormers can achieve competitive results with other state-of-the-art few-shot learning methods.

A. Broader Impact

The proposed algorithm can be applied to few-shot visual recognition fields, such as robot vision systems that must recognize new objects, camera systems that must infer the existence of new objects, and monitoring systems that must recognize new categories of objects. Since the distribution of data in the real world follows the long-tail distribution, the uncommon (lack of data) objects are common in the practical application of visual recognition system, and our algorithm can improve the robustness of visual recognition system. Although the work in this paper only focuses on the classification problem, the generalization of other tasks in visual recognition (detection, segmentation) can also be improved by using the method in this paper.

While our methods have made progress in identifying and inferring rare objects, they are still far below human level. For application scenarios with low fault tolerance, such as autonomous driving, surgery, etc., relying on the visual system's ability to correctly interpret anomalies is risky for current systems, even with the advances described in this article.

B. Implementation Details

Backbone architecture. We consider two backbones, as suggested in the literature for the purpose of fair comparisons. We resize each image patch to 84 × 84 × 3 before using the backbones.

Conv-64F. The Conv-64F [27, 32] contains 4 repeated blocks. In each block, there is a convolutional layer with 3kernel, a Batch Normalization layer, a ReLU, and a Max pooling with size 2. We set the number of convolutional channels in each block as 64. A bit different from the literature, we add a global max pooling layer at last to reduce the dimension of the embedding and get patch embeddings.

ResNet-12. We use the 12-layer residual network in [15]. The DropBlock is used in this ResNet architecture to avoid overfitting. A bit different from the ResNet-12 in [15], we apply a global average pooling after the final layer, which leads to a 640 dimensional patch embeddings.

Pre-training strategy. As mentioned before, we apply an additional pre-training strategy as suggested in [13, 32]. The backbone network, appended with a softmax layer, is trained to classify all classes in the training class split (e.g., 64 classes in the *mini*ImageNet) with the cross-entropy loss. In this stage, we apply color jitter, random crop, and random horizontal flip to increase the generalization ability of the model. After each epoch, we validate the performance of the pre-trained weights based on its few-shot classification performance on the model validation split. Specifically, we randomly sample 100 N-way 1-shot few-shot learning tasks (N equals the number of classes in the validation split, e.g., 16 in the *mini*ImageNet), which contains 1 instance per class in the support set and 15 instances per class for evaluation. Based on the penultimate layer instance embeddings of the pre-trained weights, we utilize the nearest neighbor classifiers over the few-shot tasks and evaluate the quality of the backbone. We select the pre-trained weights with the best few-shot classification accuracy on the validation set. The pre-trained weights are used to initialize the feature extractor, and the weights of the whole model are then optimized together during the meta-training stage.

Algorithm 2 Pseudo code of sparse spatial transformers layer in a PyTorch-like style

```
# key projection head $h_k$
# query projection head $h_q$
# value projection head $h_v$
Input: query feature $q$, support feature $S$
Output: aligned prototype $v_{S|q}$
    support key $\in \mathbb{R}^{N\times C'\times MK}$
    $k_S$ = $h_k(S)$
    # query query $\in \mathbb{R}^{C'\times K}$
    $q_q$ = $h_q(q)$
    # query value $\in \mathbb{R}^{C'\times K}$, support value $\in \mathbb{R}^{N\times C'\times MK}$

    $v_q$, $v_S$ = $h_v(q)$, $h_v(S)$
    # similarity matrix $\in \mathbb{R}^{N\times K\times MK}$
    $\mathbf{R}$ = $q_q$.unsqueeze(0).transpose(-1, -2) $\times$ $k_S$
    # max index $q_q^{max} \in \mathbb{R}^K$, $k_S^{max} \in \mathbb{R}^{NMK}$
    $q_q^{max}$ = $\mathbf{R}$.permute(1, 0, 2).view($K, -1$).max(-1)[1]
    $k_S^{max}$ = $\mathbf{R}$.permute(1, 0, 2).view($K, -1$).max(-2)[1]
    # mask $m \in \mathbf{R}^K$
    $m$= torch.gather($k_S^{max}$, 0, $q_q^{max}$)
    $m$ = ($m_{q|S}$ == torch.arange($K$))
    # attention map $a \in \mathbb{R}^{N\times K\times MK}$
    $a$ = $\mathbf{R}$ * $m$.unsqueeze(0).unsqueeze(-1)
    $a$ = dropout(nn.Softmax(dim=-1)($a$))
    aligned prototype feature $v_{S|q} \in \mathbb{R}^{N\times K\times C'}$
    $v_{S|q}$ = $a$ $\times$ $v_S$.transpose(-1, -2)
    return $v_{S|q}$
```

Optimization. Following the literature [32], different optimizers are used for the backbones during the model training. For the *Conv-64F* backbone, stochastic gradient descent with Adam optimizer is employed, with the initial learning rate set to be 0.1 and decay 0.1 every 10 epochs. For the ResNet, vanilla stochastic gradient descent with Nesterov acceleration is used with an initial rate of 5e-4 and decay 0.5 every 10 epochs.

C. Pseudo Code

A pseudo code of sparse spatial transformers layer in a PyTorch-like style is shown in Algorithm 1.

D. Implementation of SSFormers

We provide a PyTorch implementation of SSFormers for few-shot learning. Our code is available at `https://github.com/chenhaoxing/SSFormers`.

References

[1] Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019. 5, 6

[2] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *TPAMI*, 43(1):172–186, 2021. 1

[3] Haoxing Chen, Huaxiong Li, Yaohui Li, and Chunlin Chen. Multi-scale adaptive task attention network for few-shot learning. *arXiv preprint arXiv:2011.14479*, 2020. 1, 2

[4] Haoxing Chen, Huaxiong Li, Yaohui Li, and Chunlin Chen. Multi-level metric learning for few-shot image recognition. *arXiv preprint arXiv:2103.11383*, 2021. 1

[5] Zhengyu Chen, Jixie Ge, Heshen Zhan, Siteng Huang, and Donglin Wang. Pareto self-supervised training for few-shot learning. 2021. 6

[6] Wen-Hsuan Chu, Yu-Jhe Li, Jing-Cheng Chang, and Yu-Chiang Frank Wang. Spot and learn: A maximum-entropy patch sampler for few-shot image classification. In *CVPR*, pages 6251–6260, 2019. 1

[7] Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 6

[8] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. In *NeurIPS*, 2020. 3

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, volume 70, pages 1126–1135, 2017. 1

[10] Laleh Haghverdi, Aaron T L Lun, Michael D Morgan, and John C Marioni. Batch effects in single-cell rna-sequencing data are corrected by matching mutual nearest neighbors. *Nat. Biotechnol.*, 2018. 2, 3

[11] Fusheng Hao, Fengxiang He, Jun Cheng, Lei Wang, Jianzhong Cao, and Dacheng Tao. Collect and select: Semantic alignment metric learning for few-shot learning. In *ICCV*, pages 8459–8468, 2019. 2

[12] Ruibing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *NeurIPS*, pages 4005–4016, 2019. 2, 3, 5, 6, 8

[13] Dahyun Kang, Heeseung Kwon, Juhong Min, and Minsu Cho. Relational embedding for few-shot classification. 2021. 2, 6, 7, 9

[14] Jaekyeom Kim, Hyoungseok Kim, and Gunhee Kim. Model-agnostic boundary-adversarial sampling for test-time generalization in few-shot learning. In *ECCV*, volume 12346, pages 599–617, 2020. 6

[15] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, pages 10657–10665, 2019. 6, 9

[16] Aoxue Li, Tiange Luo, Tao Xiang, Weiran Huang, and Liwei Wang. Few-shot learning with global class representations. In *ICCV*, pages 9714–9723, 2019. 1

[17] Wenbin Li, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo. Revisiting local descriptor based image-to-class measure for few-shot learning. In *CVPR*, pages 7260–7268, 2019. 1, 2, 6

[18] Wenbin Li, Jinglin Xu, Jing Huo, Lei Wang, Yang Gao, and Jiebo Luo. Distribution consistency based covariance metric networks for few-shot learning. In *AAAI*, pages 8642–8649, 2019. 2, 6

[19] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(2):318–327, 2020. 1

[20] Yang Liu, Tu Zheng, Jie Song, Deng Cai, and Xiaofei He. DMN4: few-shot learning via discriminative mutual nearest neighbor neural network. 2021. 2

[21] Su Lu, Han-Jia Ye, and De-Chuan Zhan. Tailoring embedding function to heterogeneous few-shot tasks by global and local feature adaptors. In *AAAI*, pages 8776–8783, 2021. 6

[22] Boris N. Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. TADAM: task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, pages 719–729, 2018. 5, 6

[23] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1

[24] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018. 5

[25] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 6, 7

[26] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *CVPR*, pages 4135–4144, 2020. 5, 6, 7

[27] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, pages 4077–4087, 2017. 1, 2, 3, 5, 6, 9

[28] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, pages 1199–1208, 2018. 1, 2, 6

[29] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: A good embedding is all you need? In *ECCV*, volume 12359, pages 266–282, 2020. 8

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 2

[31] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NeurIPS*, pages 3630–3638, 2016. 1, 2, 5

[32] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*, pages 8808–8817, 2020. 2, 3, 5, 6, 9

[33] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *CVPR*, pages 12200–12210, 2020. 2, 5, 6

[34] C. Zhang, H. Li, C. Chen, Y. Qian, and X. Zhou. Enhanced group sparse regularized nonconvex regression for face recognition. *TPAMI*, 2020. to be published, doi:10.1109/TPAMI.2020.3033994. 1

[35] Hongguang Zhang, Piotr Koniusz, Songlei Jian, Hongdong Li, and Philip H. S. Torr. Rethinking class relations: Absolute-relative supervised and unsupervised few-shot learning. pages 9432–9441, 2021. 6