Learning Primitive-aware Discriminative Representations for FSL

Jianpeng Yang
Xidian University, Xian, China
20171213693@stu.xidian.edu.cn

Abstract

Few-shot learning (FSL) aims to learn a classifier that can be easily adapted to recognize novel classes, given only a few labeled examples per class. Limited data keep this task challenging for deep learning. Recent work has achieved promising classification performance, where the image-level feature from global average pooling operation is used to measure the similarity among samples. However, these global features ignore abundant local and structural information that is transferable and consistent between seen and unseen classes. How can humans easily recognize novel classes with only few samples? Some study in cognitive science argue that humans can recognize novel classes with the learned primitives. Although base and novel classes are non-overlapping, they can share some primitives in common. We expect to mine both transferable and discriminative representation from base classes and adopt them to recognize novel classes. Concretely, building on the episodic training mechanism, we propose a Primitive Mining and Reasoning Network (PMRN) to learn primitive-aware discriminative representation in an end-to-end manner for metric-based FSL model. We first add self-supervision auxiliary task in parallel, forcing model to learn visual pattern corresponding to primitives. To further mine and produce transferable primitive-aware representations, we design an Adaptive Channel Grouping (ACG) module to synthesize a set of visual primitive features from object embedding by enhancing informative channel maps while suppressing useless ones. Based on the learned primitive feature, a Semantic Correlation Reasoning (SCR) module is proposed to improve discriminative power of primitives by capturing internal relations among them. Finally, we learn the task-specific importance of primitives and conduct the primitive-level metric based on task-specific attention feature. Extensive experiments show that our method achieves state-of-the-art results on six standard benchmarks.
1. Introduction

In recent years, deep learning (DL) has achieved tremendous success in various recognition tasks with abundant labeled data[6,7,8,9,10,11]. In order to train these supervised models efficiently, we need lots of annotated images that are expensive and time-consuming to obtain, especially in the situation where a large amount of new classes are provided to recognize. Therefore, how to recognize new classes with few labeled samples has attracted more attention. To reduce the reliance on human annotation, few-shot learning (FSL) has been proposed and studied widely [1, 2, 3, 4, 5], which aims to learn a classifier that can be rapidly adapted to novel classes given just several labeled images per class.

FSL attempts to transfer the knowledge learned from base classes that have sufficient labeled data to non-overlapping novel classes with only one or a few examples. Humans can rapidly classify an object into one of several unseen classes at present by comparing the differences among them. Inspired by this ability of humans, a series of few-shot learning methods adopts metric-based methods [4, 5, 12] that learn a global image-level representation in an appropriate feature space and directly calculate the distances between the query and support images for recognition.

Nevertheless, most of these work measure similarity on image-level pooled representation for classification, which could destroy image structures and ignore local information. Local representations can provide discriminative and transferable information across classes in the few-shot scenario[15]. Due to low-data regimes in few-shot learning, the tangled background and large intra-class variations may mislead the image-level embedding from the same category far apart in a given metric space [13, 14, 15].

Therefore, local representations (LRs) based methods [16, 15, 14, 13, 17] employ low-level information to measure the distance between query images and support images. Specially, feature maps are divided into a set of local patches and the measure is conducted on these local patches. Despite they can achieve better recognition results,
local patches corresponding to small grids on the feature map just like pixel-level that do not have any interpretability, and introduce a large amount of computation and redundant background noises. Also, these methods pay more attention to how to measure on these dense feature by complex distance functions rather than how to mine discriminative and transferable feature.

Revisiting the process that humans recognize new concepts or objects, humans can first learn primitives from plenty of previously known classes [18] and then apply these primitives to learn novel classes. Specifically, humans first focus on some shared or similar primitives between known and novel classes, and then mine discriminative features from novel objects based on these primitives. Although known classes and novel classes are non-overlapping, they can share some primitives in common. In practice, primitives are viewed as object parts, or more broadly, regions capturing the compositional structure of the examples [19], which seems to have no a clear or fixed boundary. Some work on interpretability of deep networks show that CNN feature channels often correspond to some certain type of visual pattern in the input image, such as object parts [20, 21, 22]. Inspired by these works, CPDE [23] designs a soft composition mechanism, which regards each channel of the feature map as a primitive and select important primitives by enlarging the activation of them while reducing that of others during the known-class training. However, they only select top-k channels, which cannot take full advantage of informative clues from all the channels adaptively. TPMN [24] proposes a series of part filters to automatically generate part-aware representation from feature channels and utilizes part-level similarities for classification. Whereas they compose part representation separately and neglects internal semantic correlation among these part-aware representation. Compared to these methods, we thereby compose primitives by selecting feature channels related to object parts rather than single channel, which aggregates visual pattern that is coherent on semantic relation. Otherwise, it is important to note that all the local representations based methods [16, 15, 14, 13, 17, 24] lose sight of the internal semantic correlation among them. Hence, we propose to exploit the internal semantic correlation among primitives, which can improve discriminative ability of primitive-aware representation.
In this paper, we propose a Primitive Mining and Reasoning Network (PMRN) to learn discriminative primitive-aware representation for metric-based FSL model through episodic training mechanism.

For primitive mining, we first guide the feature extractor to learn visual pattern corresponding to semantic parts that can be shared between known classes and novel classes by solving jigsaw puzzles with self-supervised loss. As an auxiliary task of multi-task learning, we divide images horizontally and vertically into nine patches and disrupt the orders. The target of training is to encourage model to implicitly encoding object parts through predicting the relative location of patches. But that's not enough because patches obtained by grid dividing suffer from large randomness, which some patches just contain background noises. To further mine and aggregate transferable primitive-aware representations, we design an Adaptive Channel Grouping (ACG) module to synthesize a set of visual primitive feature from feature map. Specifically, we cluster and weight a group of feature channels that are correlated to a certain part to compose a certain primitive, which is common in known and unseen classes. Activation channels from the same primitive are also spatially-correlated, where their peak responses appear in neighboring locations. It is worth noting that parameters of ACG module are learned by episodic training mechanism aiming to obtain task-relevant primitives for few-shot learning, thereby all the primitives extracted from samples belonging to an episode are task-specific. Therefore, the ACG module can be optimized adaptively across tasks and primitives can be easily adapted to novel classes.

For primitive reasoning, we argue that mining the correlation between primitives can be more discriminative than individual primitives. Hence, a Semantic Correlation Reasoning (SCR) module is proposed to capture internal correlations among primitives. We conduct the reasoning process by constructing a graph whose nodes embedding are primitives and edges are semantic relations among primitives. Then SCR jointly and iteratively learns the semantic dependencies among nodes to guide the discriminative information propagation across primitives. After primitive reasoning, each primitive is enhanced by correlation weighted aggregating other primitives, and each primitive
encodes semantic correlation with the other primitives of an object implicitly, which make them more discriminative than before.

Intuitively speaking, humans not only focus on different regions when recognizing different novel classes from different episodes, but also pay different attention to different regions of objects when recognizing novel classes of a certain episode. That is, humans first select task-relevant regions and then devote more attention resources to significant regions for more detailed information while suppress other useless regions. Inspired by this ability of human, we propose a task-specific weighting methods that aims at weighing importance of primitives in a task and calculate image-level similarity by weighting primitive-level similarity between primitive feature of support set and query set. Concretely, a pair of primitive feature are compressed into a pair of maps and then we reconstruct a task attention feature by concatenating all pairs of primitive maps along channel dimension. Therefore, measuring the importance among channels of task attention feature can be regarded as the importance among different pair of primitives.

The main contributions of our work are summarized as follows:

(1) Building on the episodic training strategy, we propose a Primitive Mining and Reasoning Network (PMRN) to learn discriminative primitive-aware representation for metric-based FSL model, which is jointly trained and optimized in an end-to-end manner.

(2) To extract discriminative and transferable representation, we design an Adaptive Channel Grouping (ACG) module to synthesize a set of task-specific visual primitives by adaptively weighting and clustering a group of feature channels that are correlated to a certain part.

(3) We propose a novel Semantic Correlation Reasoning (SCR) module to jointly learn the semantic dependencies among primitives and guide the discriminative information propagation across primitives by constructing a graph on primitives.

(4) Inspired by human’s attention mechanism, we propose a task-specific weight module aiming at weighing importance of primitives extracted in a task, where we rec-
onstruct a novel task attention feature containing all pairs of primitive map to adaptively generate task-specific weights.

2. Related work

**Few-Shot Learning (FSL).** Most recent literature about few-shot learning mainly involve two types of research methods, metric based methods and meta-learning based methods. The meta-learning based methods [2, 36, 3, 37, 38] optimize a meta-learner utilizing the learning-to-learn paradigm [25, 26], which can rapidly adapt to new classes with just few samples for few-shot learning. [27] employs an external memory module to communicate with an LSTM-based meta learner to weight update weight. In the work [2], an LSTM-based meta-learner is designed as an optimize replacing the SGD optimizer to learn a task-specific initialization for the model. MAML [3] and some variants [31, 29, 30] aim to learn a better parameter initialization that can be quickly adapted to a novel task. The metric-based methods [32, 4, 33, 5, 12, 34, 35] learn the feature representations for input samples in an appropriate embedding space, where image-to-image similarities are calculated among different classes through diverse distance metrics. [32] fist applied the metric-based method to few-shot learning, which aimed to generalize representations to novel classes through a Siamese Neural Network. MatchNet [4] utilized episodic training mechanism and selected cosine similarity as metric to solve few-shot learning. ProtoNet [5] regarded the mean value of each class’s embedding as prototype and calculated the euclidean distance between support and query samples for classification. Our proposed PMRN belongs to metric-based methods. These methods mainly utilize a image-level representation for classification. Classification. Compared to above methods, our method adaptively mine and exploit potential local representation related to object parts, which is more consistent and transferable among the seen and unseen classes.

**Dense local Feature based FSL.** In contrast to previous methods, some FSL work [13, 16, 15, 17] focus on local representations and try to exploit the discriminative ability of local patches. Specifically, the local patch is considered as each spatial grid in the feature map and aggregate all the patch-level distance as final result. DN4 [13] introduces Naive-Bayes Nearest Neighbor into FSL and computes image-level si-
milarities via a k-nearest neighbor search over local patch. DC [25] aims to predict for each local features and calculates the average the of results as the final prediction. ATL-Net [16] proposes to adaptively select important semantic patches by an episodic attention mechanism. DeepEMD [15] conducts a many-to-many matching method among local patches via the earth mover’s distance. MCL [17] consider the mutual affiliations between the query and support features to thoroughly affiliate two disjoint sets of dense local features. Nevertheless, these methods manually preset local patch as each grid or each spacial location in the feature map, which is semantically random and do not have any interpretability. Also, dense local patches means lots of computation in the phase of measure and redundant information. Our method adaptively mine and compose spatially and semantically related visual pattern as primitives in each episode. Therefore, local representations appear in the form of several key primitives related to object parts, which is semantic naturally and leads to a better interpretability.

**Interpretability of deep networks.** Some work [22, 20, 21] show that channels of feature extracted by deep networks correspond to some certain visual patterns in the object. Such as object part, etc [20]. In this paper, inspired by these studies, we aggregate channels correspond to a certain object part as a primitive, and compute primitive-level similarity between query set and support set.

**Self-supervised learning (SSL).** Manual labels are expensive and time-consuming to collect in practice. SSL aims at learning representations from structural information without label in the object itself [39, 40]. Recently, some work [41, 23, 42] introduce SSL methods such as predicting the rotation and predicting the relative position to FSL. In this paper, we propose to encourage the backbone to learn visual pattern related to object parts by using SSL loss as a regularizer.
Figure 1. The framework of Primitive Ming and Reasoning Network (PMRN) under the 5-way 1-shot setting. We first generate global feature for support class and query sample through feature extractor $f(\cdot)$. Then the Adaptive Channel Grouping Module (ACG) automatically mines and generates a set of visual primitives for each global feature. Afterwards, the Semantic Correlation Reasoning Module (SCR) captures and reasons the internal correlation among primitives to enhance the discriminative power. Meanwhile, we generate task-specific weight for each primitive-level feature based on the task-specific attention features that are reconstructed on the primitives. Finally, we conduct primitive-level metric in place of image-level metric to calculate the final similarity for classification.

Figure 2. The illustration of self-supervised jigsaw puzzles auxiliary task. We encourage the feature extractor to help learning visual pattern corresponding to semantic parts to by adding self-supervised loss as a regularizer.
3. Method

In this section, we first introduce the general settings of few-shot learning methods. Then we explain our proposed Primitive Mining and Reasoning Network (PMRN) concretely.

3.1 Description and Formulation of few shot learning

In few-shot learning scenario, three sets of data are provided, support set $S$, query set $Q$, and an auxiliary set $B$. The given support set $S$ contains $N$ unseen classes (without train), and each class has $K$ samples. FSL aims at recognizing an unlabeled query sample $q \in Q$ into one of the $N$ support classes, and we call such a task as an $N$-way $K$-shot task. However, the support set $S$ only has several samples per class, and learning a model through just several labeled samples to classify query set must be hard. To tackle this issue, an auxiliary set $B$ is employed to train a model thorough episodic training mechanism for learning transferable knowledge. In episodic training, the auxiliary set $B$ is divided into many $N$-way $K$-shot tasks $T$ randomly (also called episodes), where each $T$ contains an auxiliary support set $B_S$ and an auxiliary query set $B_Q$, which is composed of $M$ samples per class. During the training process, the model is forced to conduct hundreds of tasks $T$ to simulate the test scenario, in order to obtain the generalization ability across tasks and to learn transferable knowledge that can be used in new $N$-way $K$-shot tasks. Note that auxiliary the set $B$ contains abundant classes and labeled samples, but it’s label space is disjoint with the set $S$ and $Q$.

The overall framework is shown in Figure 1 and Figure 2. All the images are first fed into feature extractor $f(\cdot)$ in the form of episode to acquire feature representations. Then the Adaptive Channel Grouping (ACG) module generates a set of task-specific visual primitives across tasks by adaptively weighting and clustering a group of feature channels that are correlated to a certain object part. Meanwhile, we encourage the feature extractor to learn visual pattern corresponding to semantic parts by adding self-supervised jigsaw puzzles loss as a regularizer. Afterwards, the semantic relations among primitive-level feature can be jointly learn by Semantic Correlation Reasoning (SCR) module, which constructs a graph on primitives to propagate the discriminative information across primitives and improve the discriminative of these primitive-level feature. Inspired by human’s attention mechanism, a series of task-specific weight is
adaptively learned to measure the importance of primitives for a certain task, where we reconstruct a new task-specific feature to contain all pairs of compressed primitive map, and then consider the importance among channel maps as the weight of primitive pairs. Finally, we can obtain the classification score by weighting the primitive-level similarities between support set and query set. It is worth noting that our proposed entire network is jointly trained from a scratch in an end-to-end manner without any external data.

3.2 Definition and Generation of primitives

Humans can first learn primitives [18] from known classes and then capture discriminative primitives in a new task only given few samples to compare them for classification. In addition, some research on interpretability of deep networks show that feature channels often corresponds to some certain type of visual pattern, such as object parts [20, 21, 22]. Inspired by these analysis, some work [19, 23] view object parts as primitive, or more broadly, regions capturing the compositional structure of the examples [19]. Specifically, single channel map with high activation related to object parts is selected as visual primitive [23]. Compared to these methods, our proposed methods generate primitives by aggregating feature channels related to object parts rather than single channel or part annotation. Therefore, we need to first encourage the model to learn visual pattern related to object parts. To achieve this goal, general approach [19] needs manually labeled object parts or attribute in each image to supervise the network to detect them. However, collecting so much part annotation is expensive and time-consuming. Some paper [23, 41, 42] explore how to combine the self-supervision task and FSL to reduce the reliance on part labels, which inspires us to mine primitives with the help of self-supervision loss.

self-supervision auxiliary task. In order to assist the subsequent primitive extraction operation, we fist use the self-supervision jigsaw task loss [28] for FSL as a regularizer, which force model to recognize related location of image patches by learning visual pattern corresponds to object parts. Note that self-supervision auxiliary task do not need any extra part annotation. Concretely, we first divide an input image $x \in B = \{(x_i, y_i)\}_{i=1}^{n}$ into $h \cdot w$ patches along rows and columns. Afterwards, these patches
are permuted randomly as input $x^p$ and we get index of the permutation as the target label $y^p$. The target is to predict each index of permutation for permuted patches of each image. Then, all permuted patches are fed into feature extractor $f(\cdot)$ to obtain $h \cdot w$ features and we concatenate the permuted features and use a FC layer for classification with a cross-entropy loss between the target and prediction. Let’s denote the concatenated features as $f^p$, and the classification loss can be formulated as the negative log-probability:

$$L_{ssl} = - \sum_{x \in B} \log p(f^p \mid x)$$ (1)

where $f^p \in R^{h \cdot w \cdot d}$, $h \cdot w$ is the number of patches and $d$ is the dimension of feature map. In practice, $h \cdot w$ is set as $3 \times 3$ and the number of index reach $9!$. To reduce the degree of difficulty, we reduced it to $35$ as the procedure proposed in [28], which grouped the possible permutations based on the hamming distance.

**Adaptive channel grouping module.** The above process implicitly encode visual pattern related to object parts by self-supervision auxiliary. Based on a series of spatially-correlated visual patterns that are encoded into channel activation, we design an Adaptive Channel Grouping (ACG) module to synthesize a set of visual primitive feature from these feature map adaptively.

In each episode, all the images from support set $B_S = \{(x^s_j, y^s_j) \}, j \in [1, NK]$, query set $B_Q = \{(x^q_j, y^q_j) \}, j \in [1, NM]$ are fed into feature extractor $f(\cdot)$ to obtain a collection of feature map $\Omega = \{f^s_i \}_{i=1}^{NK} \cup \{f^q_i \}_{i=1}^{NM}$, which consists of support set feature map $f^s \in R^{H \times W \times C}$ and $f^q \in R^{H \times W \times C}$ of query set. The dimension of these feature map is $H \times W \times C$, where $H$, $W$, $C$ indicate height, width, and the number of feature channels. Afterwards, a set of parallel channel grouping operations are employed to weight $k$ group of feature channels to generate $k$ primitives $P = \{p_1, p_2, \ldots, p_k \}$ for each sample, and we denote these operations as $O = \{o_1, o_2, \ldots, o_k \}$. 
Intuitively, different feature channels encode different spatially-correlated visual patterns, which contributes to different primitives. Therefore, each channel grouping operation $o_i$ is responsible to generate a series of weights for each primitive:

$$D_i = o_i(f) = [d_1, d_2, \ldots, d_c], c \in [1, C]$$  \hspace{1cm} (2)

Where the $f \in \Omega$ is feature map and $D_i$ is a set of learned weights for the $i^{th}$ primitive $p_i$. Then, we employ each set of learned weights to achieve the grouping process for each primitive by weighting all the channels, where the spatially and semantically correlated channels are assigned higher value. After the channel grouping operation $o_i$, we further obtain the primitive-level attention mask for the $i^{th}$ primitive $p_i$ as follows:

$$m_i = \sigma \left( \sum_{j=1}^{C} d_j \cdot a_j \right), i \in [1, k]$$  \hspace{1cm} (3)

where $a_j$ is the $j^{th}$ channel map of convolutional features $f \in \Omega$, and the multiplication operation here is conducted between a scalar $d_c$ and a matrix $a_c$. We use sigmoid function $\sigma(\cdot)$ on the weighted channel map to generate the $i^{th}$ primitive-level attention mask $m_i \in \mathbb{R}^{H \times W}$, which covers the activation region belonging to $i^{th}$ primitive $p_i$.

Finally, the feature map $f \in \mathbb{R}^{H \times W \times C}$ is filtered through primitive-level attention mask along channel dimension to acquire the original primitive:

$$p_i = f \otimes m_i, i \in [1, k]$$  \hspace{1cm} (4)

where $\otimes$ denotes the element wise multiplication between $m_i \in \mathbb{R}^{H \times W}$ and every channels of $f \in \mathbb{R}^{H \times W \times C}$. Note the dimension of primitive $p_i \in \mathbb{R}^{H \times W \times C}$ is the same as feature map $f \in \mathbb{R}^{H \times W \times C}$. After above operations, we capture $k$ primitive for each feature map.

Moreover, each channel grouping operation is encouraged to adaptively produce a set of weights, so we utilize global average pooling on primitive and employ two fully-connected (FC) layers for every primitives to learn primitive-specific weights. Meanwhile, we use all the images from support set and query set that belongs to a special task to train adaptive channel grouping module base on the episodic mechanism, which make the generation process of weights mix the class information involved in a special task. Hence, our proposed adaptive channel grouping module is task-specific and is more suitable for few-shot learning scenario.
In the training stage, it is worth noting that all the primitive-level attention mask tend to focus on the similar spacial and semantic region, which is usually the most discriminative regions. However, the diversity of primitive can further provide rich information and help robust recognition, especially for occlusion cases. To avoid the duplication of learned primitives, we utilize a channel grouping diversity loss inspired by [43, 24], which is formulated as:

\[
L_{\text{div}} = \sum_{i=1}^{k} \sum_{j \neq i}^{k} \delta(g(p_i), g(p_j))
\]

where \(\delta(\cdot)\) denotes the cosine similarity function between primitives from a sample, and \(g(\cdot)\) is the global average pooling operation in the spacial dimension.

As is shown in loss formulation, \(L_{\text{div}}\) will be higher if the \(i^{th}\) and \(j^{th}\) primitive represent similar spacial region and semantic feature. Therefore, we can train the module to get various primitives by minimizing the \(L_{\text{div}}\) loss. Here we conduct global average pooling on primitives rather than calculate dense pixel-level similarity based on primitive \(p_i \in R^{H \times W \times C}\) for improving training and computational efficiency.

### 3.3 Semantic Correlation Reasoning module

Assume we have obtained original transferable primitives \(P = \{p_1, p_2, \ldots, p_k\}\), the next goal is to enhance the the discriminative power of them. Compared to previous local feature based methods[23, 16, 15, 14, 13, 17, 24], the key difference is that we propose to capture and exploit the internal semantic correlation among primitives, which can be more discriminative than individual primitives. Hence, a Semantic Correlation Reasoning (SCR) module is proposed to jointly and iteratively enhance the discriminative power of primitives.

**Node embedding and adjacent matrix.** We conduct the reasoning process by constructing a graph based on learned original primitives and employ graph convolution operation. To construct the graph for mining correlation among primitives, we use \(k\) primitive feature vectors as \(k\) nodes embedding and feed them into Graph Convolution Network (GCN)[56]. We apply global average pooling operation on primitives \(P = \{p_1, p_2, \ldots, p_k\}\) to get \(k\) primitive feature vectors as node embedding \(N = \{n_1, n_2, \ldots, n_k\}, n_i \in R^C\).
Then, the adjacent matrix of correlation is computed based on node embedding which indicates internal semantic correlation between primitives. Concretely, we apply the normalized embedded Gaussian function to calculate the similarity of the two nodes as follows:

\[ S(n_i, n_j) = \frac{e^{\theta(n_j)\omega(n_i)^T}}{\sum_{j=1}^{k}e^{\theta(n_j)\omega(n_i)^T}} \]

(6)

where \( N \) is the total number of the nodes that corresponds to primitives. The dot product is used to measure the similarity of the two feature in an embedding space.

Specifically, given the input vector \( v \in \mathbb{R}^{k \times C} \) whose size is \( k \times C \), \( k \) is the number of nodes and \( C \) is the dimension of each node. We first project it into \( k \times C_e \) with two projection functions, i.e., \( \theta(\cdot) \) and \( \omega(\cdot) \), which are implemented with \( 1 \times 1 \) convolutional layer. Then two embedded feature vectors are rearranged and reshaped to an \( k \times C_e \) matrix and a \( C_e \times k \) matrix, which are multiplied to obtain an \( K \times K \) similarity matrix \( C \). The element \( C_{ij} \) of \( C \) represents the similarity of node \( n_i \) and \( n_j \), and we conduct softmax operation along the row of matrix for normalization. Based on Eq.(6), the adjacent matrix \( C \) can be calculated as follows:

\[ C = \sigma(\theta(v) \circ \omega(v)^T) \]

(7)

where \( \sigma \) is softmax operation. Through above steps, we construct a graph based on the primitives where the primitive vectors are embedded into nodes and the edges are view as similarity between nodes.

**Adaptive graph convolution layer.** Now that we obtain the adjacent matrix, an adaptive graph convolutional layer is designed to updates the node features \( v \in \mathbb{R}^{k \times C} \), which uses node features \( v \in \mathbb{R}^{k \times C} \) containing \( k \) node vectors and adjacent matrix \( C \in \mathbb{R}^{k \times k} \) as inputs. Then, the operation of **graph convolution layer** can be formulated as:

\[ v^* = G(v, C) = \varepsilon(Cvw) \]

(8)

where \( w \in \mathbb{R}^{C \times C} \) is the learned weight parameters \( 1 \times 1 \) convolution operation, and \( \varepsilon(\cdot) \) is ReLU function. Afterwards, we compose multiple graph convolution layers to Semantic Correlation Reasoning module, which jointly and iteratively reason the semantic correlation among nodes by propagating and aggregating discriminative information.
across nodes. In this way, each primitive is enhanced by weighting correlation among other primitives, and each primitive implicitly encodes semantic correlation with the other primitives of an object, which further improve the discriminative ability of primitives.

3.4 Task-specific weight module

For each sample of a task, we extract a set of transferable and discriminative primitives \( P = \{ p_1, p_2, ..., p_k \} \) through ACG and SCR module. Intuitively speaking, the importance of different primitives for a task is different. In practice, humans can first select task-relevant regions and then devote more attention resources to significant regions for more detailed information while suppress other useless regions. Inspired by this ability of humans, we propose a task-specific weighting module that aims at measuring importance of primitives in a task and calculate image-level similarity by weighting primitive-level similarity of support set and query set. For each query sample \( x^q \in B_Q \), we can extract a set of query primitives \( P^q = \{ p_1^q, p_2^q, ..., p_k^q \} \), and classify it into one of the \( N \) support classes by calculating similarity between them. For \( n^{th} \) support class, we can get \( M \) set of primitives \( P^{s,j} = \{ p_1^{s,j}, p_2^{s,j}, ..., p_k^{s,j} \} \) from \( M \) samples and we average them to get the only primitive-level representation for each class:

\[
P^s = \frac{1}{M} \sum_{j=1}^{M} P^{s,j}
\]

After that, each support class also have a a set of support primitives, which can be denoted as \( P^s = \{ p_1^s, p_2^s, ..., p_k^s \} \). Normally, the final similarity between query sample \( x^q \in B_Q \) and \( c^{th} \) support class can be calculated by summing primitive-level similarities simply, which can be formulated by:

\[
I(x^q, c) = \sum_{i=1}^{k} \phi(g(p_i^q), g(p_i^s))
\]
where $\phi(\cdot)$ is metric function to measure similarity, which is implemented as cosine similarity in this paper. and $g(\cdot)$ is global average pooling operation. As above analysis that the contribution of different primitives is discrepant, equally aggregating the primitive-level similarities makes no sense. Therefore we design a task-specific weight module to adaptively assign appropriate weight for each pair primitive $\{p^q_i, p^s_i\}_{i=1}^k$, which is responsible to compute each primitive-level similarity respectively.

Concretely, we first compress each primitive of a pair primitive $\{p^q_i, p^s_i\}$ into a map to obtain a pair of maps $\{m^q_i, m^s_i\}$. Then we concatenate them along channel dimension as follows:

$$m_i = m^q_i \oplus m^s_i, m_i \in R^{H \cdot W \cdot 2} \quad (11)$$

After that, we reconstruct a task-specific attention feature $F_a \in R^{A \cdot H \cdot W}$ by further concatenating all pairs of map $\{m^q_i, m^s_i\}_{i=1}^k$ corresponding to all pair of primitives $\{p^q_i, p^s_i\}_{i=1}^k$ and the detailed process is as Figure 2 shows.

![Figure 2](image)

Figure 2. The illustration of the task-specific weight module. Through this module, we reconstruct a task-specific attention feature $F_a$ by concatenating primitive maps. The weight generator $G(\cdot)$ further learn the task-specific weight $W \in R^k$.

In contrast to general image-level representation, the task-specific attention feature $F_a \in R^{A \cdot H \cdot W}$ contains all pairs of primitive maps $\{m^q_i, m^s_i\}_{i=1}^k$ of query set and support set, which embeds the categorical information involved in specific task. The number of channels in the $F_a \in R^{A \cdot H \cdot W}$ is $A$, and $A=2k$. Now that channels in the $F_a \in R^{A \cdot H \cdot W}$ represents related primitive map, we propose a weight generator $G(\cdot)$ based on $F_a$ to measure the importance among different primitive pair by learning the importance of
different channel maps. In this way, a set task-specific weight $W \in \mathbb{R}^k$ for each primitive pair can be generated:

$$W = \text{sigmoid}(\gamma(g(F_a), \gamma(m(F_a))))$$

(12)

where $g(\cdot)$ and $m(\cdot)$ are global average pooling and max pooling separately. $\gamma(\cdot)$ consists of two consistent $1 \times 1$ convolution layers and a sigmoid function. To sum up, we train the weight generator $G(\cdot)$ across tasks with amounts of task-specific attention feature $F_a$ to adaptively assign higher weights to significant primitives that means more contribution to recognition. Therefore, the final global similarity should be change to the weighted sum of primitive-level similarities:

$$I(x^q, c) = \sum_{i=1}^{k} W_i \phi(g(p_i^q), g(p_i^s))$$

(13)

3.4 Loss and training

Based on episodic training mechanism, all the modules in our proposed Primitive Mining and Reasoning Network (PMRN) are jointly trained end-to-end from scratch without any extra data. Moreover, our framework incorporates self-supervision to few-shot learning algorithm by adding an auxiliary self-supervised loss.

For few-shot learning, the final similarity $I$ between query sample $x^q$ and class $c$ can be calculated by the weighted sum of primitive-level similarities. Hence, the probability that each query sample $x^q \in Q = \{(x_j^q, y_j^q)\}, j = 1, 2, \cdots NM$ could be classified in class $c$ can be formulated as:

$$p(y_c | x^q) = \frac{\exp(I(x^q, c))}{\sum_{c'=1}^{N} \exp(I(x^q, c'))}$$

(14)

We compute the classification probability by using a softmax operation and then the cross-entropy loss is selected as the few-shot learning loss:

$$L_{cls} = \sum_{x^q \in Q} -\log(p(y_q | x^q))$$

(15)

Then, consisting of $L_{cls}$, $L_{ssl}$ and $L_{div}$, the loss of our proposed PMRN network can be defined as:

$$L = L_{cls} + \lambda L_{div} + \alpha L_{ssl}$$

(16)
where \( \lambda \) and \( \alpha \) are hyper-parameter that controls the importance of the self-supervision loss \( \mathcal{L}_{ssl} \) and diversity loss respectively.

### 4. Experiments

In this section, we first introduce the datasets involved in our experiments and then present some key implementation details. Afterwards, we compare our methods with the state-of-the-art methods on general few-shot learning datasets and fine-grained few-shot learning datasets respectively. Finally, we show some ablative experiments to validate each module in our network.

#### 4.1 Dataset Description

To evaluate the performance of our proposed methods, we conduct expensive experiments on a widely-used FSL dataset and five fine-grained datasets:

- **miniImageNet** [4] consists of 100 classes with 600 images selected from the ILSVRC-2012 [50]. We follow the split utilized by [2] and take all the classes into 64, 16 and 20 classes as train set, validation set and test set separately.

- **Caltech-UCSD Birds-200-2011** [44] contains 11,788 images from 200 bird classes. Following the splits in [49], we divide them into 100/50/50 classes for train/val/test and each image is first cropped to a human-annotated bounding box.

- **Stanford Dogs** [47] contains 120 categories of dogs with a total number of 20,580 images. **Stanford Cars** [45] contains 196 classes of cars and 16, 185 images.

- **FGVC aircrafts** [46] has 100 categories about aircrafts and 10,000 images are provided.

- **Oxford flowers** [48] consists of 102 classes about flowers with 8189 images.

The detailed number of classes and various data splits are denoted Table 1.

| Dataset            | \( N_{all} \) | \( N_{train} \) | \( N_{val} \) | \( N_{novel} \) |
|--------------------|---------------|-----------------|---------------|----------------|
| miniImageNet       | 100           | 54              | 16            | 20             |
| Caltech-UCSD Birds | 200           | 100             | 50            | 50             |
| Stanford Dogs      | 120           | 60              | 30            | 30             |
| Stanford Cars      | 196           | 98              | 49            | 49             |
| FGVC aircrafts     | 100           | 50              | 25            | 25             |
| Oxford flowers     | 102           | 51              | 26            | 25             |
Table 1. The splits of all six datasets. Nall is the number of all classes. Ntrain, Nval and Novel denotes the number of classes in training set, validation set and test set.

4.2 Implementation Details

We select the widely-used ResNet-18 [51] as the backbone of our feature extractor \(f(\cdot)\), and then remove the last pooling layer of it. The model is learned by episodic training mechanism. Each episode consists of an \(N\)-way \(K\)-shot task and 16 query samples are provided for each class. Specifically, there are 5 support images and 80 query images for 5-way and 1-shot setting while 25 support images and 80 query images for 5-way and 5-shot setting in a single episode. We train the network by ADAM [52] with a learning rate of 0.001. Note that the number of episodes is 100,000 for 5-way and 5-shot setting and 300,000 for 5-way and 1-shot setting. Our model is implemented with the Pytorch [53] based on the codebase for few-shot learning denoted in [51].

In the testing stage, we randomly sample 1000 episodes from the novel set and use the top-1 mean accuracy as the evaluation criterion. We report the final mean accuracy with the 95% confidence intervals. It is worth noting that all our the modules are trained from scratch in an end-to-end manner and do not need fine-tuning in the test stage.

We use the data augmentation procedure employed in [51] which achieves a good performance. For self-supervision task, we first randomly crop the original images to get a \(255 \times 255\) region with random scaling between \([0.5, 1.0]\). Then we split it into \(3 \times 3\) regions, which contains nine random patches of size \(64 \times 64\). The number of primitives is set as on all the datasets and default value of hyper-parameter \(\lambda\) and \(\alpha\) is set as 0.4 and 1.0 separately.

4.3 General Few-shot Classification Results

We show our experiments results for general few-shot learning methods on miniImageNet. Table 2 show the comparisons of our proposed PMRN with general few-shot learning methods, including global feature based methods [7, 14, 19, 25, 34] as well as local feature based methods [16, 33, 35].
Comparison with methods based on local representations. As our method belongs to the metric-learning branch based on local representations, we first compare our method with some popular metric based methods that exploit local features. The detailed results can be seen in Table 1. In short, our method outperforms all of these methods, including MCL[17], DN4[13], ATL-Net[16], DC[25] and DeepEMD[15]. Note that our PMRN achieves amazing margin of 7.56% than the best local-based method MCL[17] in 5-way 5-shot setting and 12.69% in 5-way 1-shot. The key difference is that these recent local-based methods directly utilize the grid representations divided from the feature map, where such local patches may contain less informative clues, that is, too much randomness and background. In other word, they destroy semantic consistency of local features. However, our PMRN can adaptively mine and exploit significant visual primitives related to object parts and capture internal correlation to enhance the discriminative power of primitives. The significant accuracy gain proves that our primitive-level representations are more effective than previous local patches.

Comparison with the state-of-the-arts. We also compare PMRN with some state-of-the-art methods on minIImageNet. As Table 2 shows, our proposed PMRN achieves the new state-of-the-art performance on all settings (5-way 5-shot and 5-way 1-shot). Compared with the best method HCT, we achieve a remarkable 5.92% performance gains in 5-way 1-shot setting and 2.84% in 5-way 5-shot setting. Through the comparison with the state-of-the-arts methods, we can conclude that our proposed transferable and discriminative primitive-level representation and task-specific weight mechanism improve performance greatly in general few-shot learning.

4.4 Fine-grained Few-shot Classification Results

To further demonstrates the effectiveness of our method, we conduct expensive experiments on various fine-grained datasets for fine-grained few-shot learning methods. Table 3-Table 5 show the comparisons of our proposed PMRN with state-of-the-art methods on five fine-grained datasets.

It can be seen that our method achieves the new state-of-the-art performance under all setting. Compared with best methods on Caltech-UCSD Birds-200-2011, Stanford Dogs and Stanford Cars, our method has 3.89%, 24.48% and 14.76% accuracy
gains under the 5-way 1-shot setting and 1.64%, 18.89% and 2.49% accuracy gains under the 5-way 5-shot setting. In addition, we follow the same setting as in [41] to train some baselines from scratch on Oxford Flowers and FGVC Aircraft. Compared with these baselines, we also achieve remarkable performance gains on all setting. Note that our proposed PMRN achieve larger performance gains on fine-grained datasets than general few-shot learning datasets. For fine-grained few-shot learning, the image-level representation can not capture discriminative fine-grained information due to the smaller inter-class and larger intra-class variations. However, our methods adaptively mine and enhance primitives with abundant local information, which can be more effective to distinguish fine-grained classes.

| Method                  | Backbone    | miniImageNet |
|-------------------------|-------------|--------------|
|                         |             | 5-way 1-shot | 5-way 5-shot |
| MatchingNet[4,55,15]    | ResNet-12   | 65.64        | 78.72        |
| RelationNet[12,17]     | ResNet-12   | 60.97        | 75.32        |
| ProtoNet[5,17]         | ResNet-12   | 62.67        | 77.88        |
| CAN[57]                | ResNet-12   | 63.85        | 79.44        |
| DN4[13]                | ResNet-12   | 65.35        | 81.10        |
| DeepEMD[15]            | ResNet-12   | 65.91        | 82.41        |
| ATL-Net[16]            | ConvNet     | 54.30        | 73.22        |
| DSN[59]                | ResNet-12   | 62.64        | 78.83        |
| FRN[58]                | ResNet-12   | 66.45        | 82.83        |
| CPDE[23]               | ResNet-18   | 65.55        | 80.66        |
| TPMN[24]               | ResNet-12   | 67.64        | 83.44        |
| CTM[60]                | ResNet-18   | 64.12        | 80.51        |
| MCL[17]                | ResNet-12   | 67.85        | 84.47        |
| EASY3-R[62]            | 3xResNet-12 | 71.75        | 87.15        |
| HCT[61]                | 3xTransformers | 74.62      | 89.19        |
| PMRN (ours)            | ResNet-18   | **80.54**    | **92.03**    |

Table 2. Comparison of our method with the state-of-the-art methods. Few-shot classification (%) results with 95% confidence intervals on miniImageNet.
### Table 3. Comparison of our method with the state-of-the-art few-shot learning methods on fine-grained dataset. The results with 95% confidence intervals on Caltech-UCSD Birds.

| Method               | Backbone | 5-way 1-shot | 5-way 5-shot |
|----------------------|----------|--------------|--------------|
| MatchNet[51,4]       | ResNet-18| 73.49        | 84.45        |
| MatchNet[49,4,15]    | ResNet-12| 71.87        | 85.08        |
| RelationNet[51,12]   | ResNet-18| 68.58        | 84.05        |
| RelationNet[12,51]   | ResNet-34| 66.20        | 82.30        |
| ProtoNet[51,5]       | ResNet-18| 72.99        | 86.64        |
| MAML[3,51]           | ResNet-18| 68.42        | 83.47        |
| SCA + MAML++ [63]    | DenseNet | 70.33        | 85.47        |
| Baseline[51]         | ResNet-18| 65.51        | 82.85        |
| Baseline++ [51]      | ResNet-18| 67.02        | 83.58        |
| S2M2[64]             | ResNet-18| 71.43        | 85.55        |
| DeepEMD[15]          | ResNet-12| 75.65        | 88.69        |
| DEML[65]             | ResNet-50| 67.28        | 83.47        |
| ATL-Net[16]          | ConvNet  | 60.91        | 77.05        |
| Cosine classifier[51]| ResNet-18| 72.22        | 86.41        |
| DSN[67]              | ResNet-12| 80.80        | 91.19        |
| FRN[58]              | ResNet-12| 83.16        | 92.59        |
| CPDE[24]             | ResNet-18| 80.11        | 89.28        |
| CFA[66]              | ResNet-18| 73.90        | 86.80        |
| MCL[17]              | ResNet-12| 85.63        | 93.18        |
| PMRN (ours)          | ResNet-18| 89.25        | 94.82        |

### Table 4. Comparison of our method with the state-of-the-art few-shot learning methods on fine-grained dataset. The results with 95% confidence intervals on Stanford Cars and Stanford Dogs.

| Method               | Stanford Cars | Stanford Dogs |
|----------------------|---------------|---------------|
|                      | 5-way 1-shot  | 5-way 5-shot  | 5-way 1-shot  | 5-way 5-shot  |
| MatchNet[4,68]       | 34.80         | 44.70         | 35.80         | 47.50         |
| GNN[69,68]           | 55.85         | 71.25         | 46.98         | 62.27         |
| DN4[13]              | 61.51         | 89.60         | 55.85         | 63.51         |
| ProtoNet[5,68]       | 40.90         | 52.93         | 40.90         | 48.19         |
| MAML[3,71]           | 47.22         | 61.21         | 44.81         | 58.68         |
| RelationNet[12,71]   | 47.67         | 60.59         | 43.33         | 55.23         |
| MML[72]              | 72.43         | 91.05         | 59.05         | 75.59         |
| ATL-Net[16]          | 67.95         | 89.16         | 54.49         | 73.20         |
| CovaMNet[68]         | 56.65         | 71.33         | 49.10         | 63.04         |
| PABN + cpt[70]       | 54.44         | 67.36         | 45.65         | 61.24         |
| LRPABN + cpt[70]     | 60.28         | 73.29         | 45.72         | 60.94         |
| MATANets[71]         | 73.15         | 91.89         | 55.63         | 70.29         |
| PMRN (ours)          | 87.91         | 94.38         | 80.11         | 89.18         |
Table 5. Comparison of our method with the some baselines methods on fine-grained dataset. The results with 95% confidence intervals on Oxford Flowers and FGVC Aircraft. We re-implement above five methods with ResNet-18 in these two datasets for fair comparison.

### 4.4 Ablation Study

To assess the effectiveness of each module in our methods, we conduct detailed ablation studies on the miniImageNet dataset. At first we introduce our baseline as the basis reference for validation study of other modules. Specifically, we use the ResNet-18 as the backbone and remove the last pooling layer. Then we use episodic training mechanism to classify each query set sample into one of the $N$ support classes called $N$-way $K$-shot task as the ProtoNet[5] shows. The difference is that we change the similarity function from euclidean distance to cosine similarity.

Table 6. Ablation results on miniImageNet in 5-way 1-shot and 5-way 5-shot settings for the proposed PMRN.

As shown in Table 6, we add various modules on the baseline respectively to verify the effectiveness of each module. Compared to the baseline methods, our proposed ACG module improve the accuracy by 2.51% in 1-shot setting and 4.06% in 5-shot setting. This large improvement prove that the original primitive-level representations
mined by ACG module possess better discriminative power. Afterwards, based on the original primitive-level representations produce by ACG module, we utilize SRC and task-specific weight module (TWA) separately to enhance the model. With the application of SCR module, further improvements by 2.87 in 5-shot setting indicates that considering the internal correlation among primitives is necessary. Meanwhile, TWA achieve a large accuracy gains of 5.25% and 7.52% in 5-shot and 1-shot setting respectively on the top of ACG module, which reveals that our proposed task-specific weight generation mechanism can select both transferable and discriminative primitives across different task indeed.

It is worth noting that the combination of SCR and TWA acquire remarkable performance improvement than any single addition of them. The 9.86% and 13.09% performance gains in 5-shot and 1-shot can adequately demonstrate they can mutually be optimized and enhanced in the training stage.

Moreover, the addition of self-supervision auxiliary task (SSL) boosts the performance base on the combination of ACG, SRC and TWA modules. As we can observe, self-supervision auxiliary task also contributes to some performance gains compared with the single baseline method. However, the performance gains of SSL on the combination of ACG, SRC and TWA modules are much larger than on the single baseline, which further confirm that the self-supervision auxiliary task assists our model to mine primitives related to objects by encouraging feature extractor $f(\cdot)$ to learn relative location of image patches. In other words, our proposed PMRN matches well with SSL due to the consistent goal on mining and capturing object parts.

4.5. Cross-Domain Experiments

Cross-Domain test is a challenging task for few-shot learning thanks to the large domain gap between different datasets, which can evaluate the model’s ability of transferring knowledge and generalization.

We employ the setting in [5], and conduct a cross-domain experiment where our model is trained on miniImagenet and evaluated on the CUB dataset. The detailed results are shown in the Table. 7. As the results show, our PMRN improves these baseline models observably under all setting, especially in 1-shot setting, e.g., 6.12% This demonstrates that our method can mine and capture transferable primitive-level local representation across domains, which encodes abundant discriminative informati-
on. Meanwhile, the internal semantic correlation among visual primitives is fully exploited to enhance the discriminative power of them. Furthermore, the task-specific weight generation mechanism guides the model to automatically learn significant visual primitives that possess remarkable discriminative ability for recognition across task. Hence, our model can be rapidly applied to the novel classes with domain gap.

| Method                  | 5-way 1-shot | 5-way 5-shot |
|-------------------------|--------------|--------------|
| cosmax [7]              | 43.06        | 64.38        |
| ProtoNet [5]            | 47.51        | 67.96        |
| centroid [54]           | 46.85        | 70.37        |
| FEAT [55]               | 50.67        | 71.08        |
| MatchingNet [4]         | 51.65        | 69.14        |
| Cosine classifier [7]   | 44.17        | 69.01        |
| TPMN [24]               | 52.83        | 72.69        |
| **PMRN**                | **58.95**    | **72.89**    |

Table 7. Cross-Domain experiments results from miniImageNet to CUB.

5. Conclusion

Inspired by the humans, we research the few-shot learning based on visual primitive, which can be viewed as object parts, or more broadly, regions capturing the compositional structure of the examples. We propose a Primitive Mining and Reasoning Network for FSL. We adaptively mine and extract primitive-level representations and conduct primitive-level metric on these primitives in a meta-learning way. Compared to image-level representation, primitives show remarkable transferable and discriminative power across tasks after internal correlation reasoning and task-specific weight operation. Note that a auxiliary self-supervision jigsaw solving task implicitly learning patches related to objects also promotes the process of primitive mining. Extensive experiments indicate the effectiveness of our methods.

Acknowledgement

Our experiments were performed on the GPU server provided by Xidian University of China.
References

[1] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[2] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

[4] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching Networks for one-shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[5] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[9] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2016.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] Tianzhu Zhang, Changsheng Xu, and Ming-Hsuan Yang. Learning multi-task correlation particle filters for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2):365–378, 2019.

[12] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HSTorr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[13] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7260–7268, 2019.

[14] Yann Lifchitz, Yannis Avrithis, Sylvaine Picard, and Andrei Bursuc. Dense classification and implanting for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9258–9267, 2019.

[15] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Differentiable earth mover’s distance for few-shot learning. *arXiv preprint arXiv:2003.06777*, 2020.

[16] Chuanqi Dong, Wenbin Li, Jing Huo, Zheng Gu, and Yang Gao. Learning task-aware local representations for few-shot learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 716–722, 2020.

[17] Yang Liu, Weifeng Zhang, Chao Xiang, Tu Zheng et al. Learning to Affiliate: Mutual Centralized Learning for Few-shot Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 14411–14420, 2022.

[18] Donald D Hoffinan and Whitman A Richards. Parts of recognition. *Cognition* 18, 1-3 (1984), 65–96.

[19] Pavel Tokmakov, Yu-Xiong Wang, and Martial Hebert. Learning compositional representations for few-shot recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6372–6381, 2019.
[20] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6541–6549, 2017.

[21] Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8730–8738, 2018.

[22] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2921–2929, 2016.

[23] Zou, Y., Zhang, S., Chen, K., Tian, Y., Wang, Y., & Moura, J. M. Compositional few-shot recognition with primitive discovery and enhancing. In Proceedings of the 28th ACM International Conference on Multimedia, pages 156–164, 2020.

[24] Jianmin Wu, Tianzhu Zhang, Yongdong Zhang, and Feng Wu. Task-aware part mining network for few-shot learning. In Proceedings of the IEEE International Conference on Computer Vision, pages 8433–8442, 2021.

[25] S. Thrun and L. Pratt. Learning to learn: Introduction and overview. In Learning to learn, pages 3–17. Springer, 1998.

[26] R. Vilalta and Y. Drissi. A perspective view and survey of meta-learning. Artificial intelligence review, 18(2):77–95, 2002.

[27] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap. Meta-learning with memory-augmented neural networks. In International Conference on Machine Learning, pages 1842–1850, 2016.

[28] Noroozi, M., Favaro, P. Unsupervised learning of visual representations by solving jigsaw puzzles. In European Conference on Computer Vision, pages 69–84, 2016.

[29] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. In International Conference on Learning Representations, 2018.

[30] Mengting Chen, Yuxin Fang, Xinggang Wang, Heng Luo, Yifeng Geng, Xinyu Zhang, chang Huang, Wenyu Liu, and Bo Wang. Diversity transfer network for few-shot learning. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 10559–10566, 2020.

[31] Qianru Sun, Yao Yao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 403–412, 2019.

[32] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In International Conference on Machine Learning Workshop, volume 2, 2015.

[33] E. Triantafillou, R. Zemel, and R. Urtasun. Few-shot learning through an information retrieval lens. In Advances in Neural Information Processing Systems, pages 2255–2265, 2017.

[34] V. Garcia and J. Bruna. Few-shot learning with graph neural networks. In International Conference on Learning Representations, 2018.

[35] W. Li, J. Xu, J. Huo, L. Wang, G. Yang, and J. Luo. Distribution consistency based covariance metric networks for few-shot learning. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 8642–8649, 2019.

[36] Q. Cai, Y. Pan, T. Yao, C. Yan, and T. Mei. Memory matching networks for one-shot image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4080–4088, 2018.

[37] S. Gidaris and N. Komodakis. Dynamic few-shot visual learning without forgetting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4367–4375, 2018.

[38] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap. Meta-learning with memory-augmented neural networks. In International Conference on Machine Learning, pages 1842–1850, 2016.

[39] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 8545–8552, 2019.

[40] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In European Conference on Computer Vision, pages 69–84, 2016.
[41] Su J C, Maji S, Hariharan B. When does self-supervision improve few-shot learning? In European conference on computer vision, pages 645–666, 2020.

[42] S. Gidaris, A. Bursuc, N. Komodakis, P. Pérez & M. Cord. Boosting few-shot visual learning with self-supervision. In Proceedings of the IEEE international conference on computer vision, pages 8059–8068, 2019.

[43] Zheng H, Fu J, Mei T, & Luo J. Learning multi-attention convolutional neural network for fine-grained image recognition. In Proceedings of the IEEE international conference on computer vision, pages 5209–5217, 2017.

[44] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltechucsd birds 200. 2010.

[45] J. Krause, M. Stark, Deng J. Fei-Fei L. 3D object representations for fine-grained categorization. In Proceedings of the IEEE international conference on computer vision workshops, pages 554–561, 2013.

[46] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, A. Vedaldi. Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151, 2013.

[47] A. Khosla, N. Jayadevaprakash, Yao B, Fei-Fei L. Novel dataset for fine-grained image categorization. In First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[48] M. E. Nilsback, A. Zisserman. A visual vocabulary for flower classification. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 1447–1450, 2006.

[49] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Learning embedding adaptation for few-shot learning. arXiv preprint arXiv:1812.03664, 2018.

[50] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhifeng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

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

[52] Wei Yu Chen, Yencheng Liu & Zsolt Kira, Jiabin Huang, et al. A closer look at few-shot classification. In International Conference on Learning Representation, 2019.

[53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[54] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high performance deep learning library. In Conference on Neural Information Processing Systems, pages 8024–8035, 2019.

[55] Arman Afrasiyabi, Jean-François Lalonde, and Christian Gagne. Associative alignment for few-shot image classification. In European Conference on Computer Vision, pages 18–35. Springer, 2020.

[56] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few shot learning via embedding adaptation with set-to-set functions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8808–8817, 2020.

[57] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations, 2017.

[58] Ruibing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In Advances in Neural Information Processing Systems 2019, 32.

[59] Davis Wertheimer, Luming Tang, and Bharath Hariharan. Few-shot classification with feature map reconstruction networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8012–8021, 2021.

[60] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In IEEE Conference on Computer Vision and Pattern Recognition, pages 4135–4144, 2020.
[60] Hongyang Li, David Eigen, Samuel Dodge, Matthew Zeiler, and Xiaogang Wang. Finding task-relevant features for few shot learning by category traversal. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[61] He, Y., Liang, W., Zhao, D., Zhou, H. Y., Ge, W., Yu, Y., & Zhang, W. (2022). Attribute Surrogates Learning and Spectral Tokens Pooling in Transformers for Few-shot Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9119–9129, 2022.

[62] Bendou, Y., Hu, Y., Lafargue, R., Lioi, G., Pasdeloup, B., Pateux, S., & Gripon, V. EASY: Ensemble Augmented-Shot Y-shaped Learning: State-Of-The-Art Few-Shot Classification with Simple Ingredients. *arXiv preprint arXiv:2201.09699*, 2022.

[63] Antreas Antoniou and Amos Storkey. Learning to learn by Self-Critique. *arXiv preprint arXiv:1905.10295*, 2019.

[64] Puneet Mangla, Mayank Singh, Abhishek Sinha, Nupur Kumari, Vineeth N Balasubramanian, and Balaji Krishnamurthy. Charting the Right Manifold: Manifold Mixup for Few-shot Learning. *arXiv preprint arXiv:1907.12087*, 2019.

[65] Fengwei Zhou, Bin Wu, and Zhenguo Li. Deep meta learning: Learning to learn in the concept space. *arXiv*, 1802.03596, 2018.

[66] Ping Hu, Ximeng Sun, Kate Saenko, and Stan Sclaroff. 2019. Weakly-supervised Compositional Feature Aggregation for Few-shot Recognition. *arXiv preprint arXiv:1906.04833*, 2019.

[67] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspace for few-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4136–4145, 2020.

[68] Wenbin Li, Jinglin Xu, Jing Huo, Lei Wang, Yang Gao, and Jiebo Luo. Distribution consistency based covariance metric networks for few-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8642–8649, 2019.

[69] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018.

[70] Huaxi Huang, Junjie Zhang, Jian Zhang, Jingsong Xu, and Qiang Wu. Low-rank pairwise alignment bilinear network for few-shot fine-grained image classification. *arXiv preprint arXiv:1908.01313*, 2019.

[71] Chen, H., Li, H., Li, Y., & Chen, C. Multi-scale adaptive task attention network for few-shot learning. *arXiv preprint arXiv:2011.14479*, 2020.

[72] Chen, H., Li, H., Li, Y., & Chen, C. Multi-level metric learning for few-shot image recognition. *arXiv preprint arXiv:2103.11383*, 2021.
