COMPAS: Representation Learning with Compositional Part Sharing for Few-Shot Classification

Ju He  Adam Kortylewski  Alan Yuille  
Johns Hopkins University

Abstract

Few-shot image classification consists of two consecutive learning processes: 1) In the meta-learning stage, the model acquires a knowledge base from a set of training classes. 2) During meta-testing, the acquired knowledge is used to recognize unseen classes from very few examples. Inspired by the compositional representation of objects in humans, we train a neural network architecture that explicitly represents objects as a set of parts and their spatial composition. In particular, during meta-learning, we train a knowledge base that consists of a dictionary of part representations and a dictionary of part activation maps that encode frequent spatial activation patterns of parts. The elements of both dictionaries are shared among the training classes. During meta-testing, the representation of unseen classes is learned using the part representations and the part activation maps from the knowledge base. Finally, an attention mechanism is used to strengthen those parts that are most important for each category. We demonstrate the value of our compositional learning framework for a few-shot classification using miniImageNet, tieredImageNet, CIFAR-FS, and FC100, where we achieve state-of-the-art performance.

1. Introduction

Advances in the architecture design of deep convolutional neural networks (DCNNs) [14, 28, 9] increased the performance of computer vision systems at image classification enormously. However, in practice, their performance is usually limited when not enough labeled data is available. Few-shot classification is concerned with the problem of learning from a small number of samples. In particular, it consists of two consecutive learning processes: 1) In the meta-learning stage, the model acquires a knowledge base from a set of training classes. 2) During meta-testing, the acquired knowledge is used to recognize unseen classes from very few examples. Hence, few-shot classification wants to emulate human learning efficiency by requiring to transfer the knowledge gained through training on a large number of base classes to enhance the learning of new classes from just a few classes.

Various approaches to few-shot classification were proposed in the past that take different perspectives on the same problem: 1) Metric-based methods aim to learn a task-invariant metric [33, 29, 31, 21]. 2) Meta-learning methods approach the problem from the perspective of learning to learn. They aim to design deep models such that the model parameters can be adapted to include new classes by optimizing very few gradient steps only on a small set of data [5, 20, 15, 36]. 3) Large-corpus-based methods directly train a model on the large base training dataset through a proxy task to get a strong and discriminative representation [6, 8, 23, 7, 32]. While these methods all try to share the common knowledge among base classes and novel classes, they do not explicitly consider that objects can have similar parts and shapes that can be reused.

In this paper, we introduce a novel approach to few-shot classification that explicitly exploits that object parts and their spatial activation patterns are shared among different object classes. During meta-learning, we train a knowledge base that consists of a dictionary of part representations and
a dictionary of part activation maps (Figure 1). We start by extracting the feature representations of an image up to the last convolution layer of a standard backbone architecture, such as ResNet [9]. Following recent work on unsupervised part detection [16, 39, 40], the part dictionary is learned by clustering the individual feature vectors from the feature encoding of the training images. Moreover, we extract part activation maps by computing the spatial activation pattern of parts in the training images. The part activation maps are clustered to learn a dictionary of prototypical maps that encode the most common spatial activation patterns of parts. In practice, the elements of the map dictionary are optimized to be distinct from each other to avoid redundancies. During meta-testing, our model learns representations of objects by composing them from the parts and part activation maps of the knowledge base. We use an attention layer to increase the weight of parts that are most discriminative for an object class. Finally, the learned object representations are fed into a classifier to predict the class label. During meta-training, the full model pipeline is trained end-to-end. However, during meta-testing, we observe that it is sufficient to train the classification head only while freezing the learned backbone and knowledge base. This is different from the majority of other meta-learning methods and highlights the strong generalization performance induced by integrating compositional part sharing into neural networks.

We evaluate our model extensively on four popular few-shot classification datasets (miniImageNet [33], tieredImageNet [26], CIFAR-FS [1], Fewshot-CIFAR100 [21]) and achieve state-of-the-art performance on all datasets. In summary, we make several important contributions in this work:

1. We introduce COMPAS, a novel neural architecture for few-shot classification that learns through compositional part sharing. To the best of our knowledge, we are the first to study how the spatial distributions of different parts as shared knowledge on few-shot classification and verify its effectiveness.

2. We propose a novel and simple method to utilize the common spatial distribution across different classes to solve the task of few-shot classification.

3. We achieve SOTA performance on several standard benchmarks, which suggests that by learning good representations, we can still outperform many recent complex optimization methods.

2. Related Work

In this section, we review existing work on few-shot classification and compositional models.

2.1. Few-shot learning

Few-shot learning has received a lot of attention over the last years. Related work can be roughly classified into three branches: Metric-based, optimization-based, and large-corpus-based learning frameworks. We will briefly review all three of them in the following.

The core idea in metric-based few-shot learning work is related to nearest neighbor algorithms and kernel density estimation. These methods embed input data into a fixed embedding space and use these to design proper kernel functions. Matching Networks [33] employed two networks for support and query samples, respectively, followed by an LSTM with read-attention to encode the full embedding. The prototypical network [29] was designed to compare the query with class prototypes in the support set by using the euclidean distance to prototype representations. Relation Networks [31] leveraged relational module to learn the metric by the network used for evaluating distance instead of manually designed. TADAM [21] proposed metric scaling and metric task conditioning to further boost the performance of Prototypical Networks.

Another branch of methods is optimization-based. Since traditional deep learning models are not designed to train with very few samples, these methods aim to find a good initialization point or update path that can lead to a quick convergence. MAML [5] proposed a general optimization algorithm that can get big improvements on a new task with a small number of gradient steps. Reptile [20] simplified MAML by removing the re-initialization for each task. MetaOptNet [15] replaced the linear predictor with an SVM in a MAML framework and introduced a differentiable quadratic programming solver to allow end-to-end training. FEAT [36] proposed set-to-set functions for a quick adaptation from instance embeddings to the target embeddings.

While many metric-based and optimization-based works tend to solve the problem in a meta-learning way, works on large-training-corpus methods argue that training a base network on the whole training set directly is also feasible. These methods can provide stronger feature representations, thus making the model more discriminative even with few samples. Dynamic Few-shot [7] extended object recognition systems with an attention weight generator and redesigned the classifier module as the cosine similarity function. Qiao et al. [23] proposed to adapt the parameters trained on meta-training sets to meta-testing sets by predicting parameters from activations. Recently, RFS [32] showed that simply training the embedding function on the combined meta-training sets followed by knowledge distillation further improves the performance. It proves that learning a good representation through a proxy task, such as image classification, can give state-of-the-art performances.

Though all these methods improve few-shot learning in
different ways, they do not explicitly take into account that objects can have similar parts and shapes which can be reused. The work closest related to ours is the Attentive prototype [34], which introduced capsule networks to account for the lack of part representations in standard deep networks. However, they do not explicitly exploit part information, and their performance is limited by the fact that capsule networks are not robust to background noise.

2.2. Compositional models

Many traditional works on compositional models for classification [2, 4, 35, 41] aimed at learning the model parameters directly from image pixels. The major challenge for these approaches is that they need to explicitly account for nuisances such as illumination and deformation. Several works have recently proposed integrating the features of higher layers of deep convolutional neural networks with compositional models since these features have shown to be robust to nuisances and have some semantic meanings.

Liao et al. [16] proposed to integrate compositionality into DCNNs by regularizing the feature representations of DCNNs to cluster during learning. Their qualitative results show that the resulting feature clusters resemble detectors of different parts. Zhang et al. [39] demonstrated that part detectors emerge in DCNNs by restricting the activations in feature maps to have a localized spatial distribution. Kortylewski et al. [13] proposed to learn generative dictionary-based compositional models from the features of a DCNN. They use their compositional model as “backup” to an independently trained DCNN if the DCNNs classification score falls below a certain threshold. Kortylewski et al. [12] further proposed a fully differentiable compositional model for image classification that shows strong robustness to either synthetic occlusion or real occlusion scenes. Sun et al. [30] demonstrated that these methods could be extended to some upper-level tasks by introducing compositional priors.

These recent advances inspire our work in integrating compositional models and deep neural networks. In particular, our model for few-shot classification learns part representations and composing them together into a whole object representation. We exploit that parts and their spatial activation patterns can be shared among different classes, which enables our model to learn efficiently from very few examples.

3. Method

We briefly review the framework of few-shot classification in section 3.1. We present how we learn the part dictionary module in section 3.2, followed by a discussion on how to learn the map dictionary module and how to integrate these modules into a pipeline for few-shot classification in section 3.3. We discuss how to train our model in an end-to-end manner in section 3.4.

3.1. Few-Shot Classification

Few-shot image classification consists of two consecutive learning processes: 1) In the meta-learning stage, the model acquires a knowledge base from a set of training classes. 2) During meta-testing, the acquired knowledge is used to recognize unseen classes from very few examples. During meta-training, a meta-task is represented as a N-way-K-shot classification problem. In this meta-task a training set is sampled with N classes and K example images per class $I^{train} = \{I_1^1, \ldots, I_N^K; y_1^1, \ldots, y_N^K\}$. This training set is often also referred to as the support set. Moreover, a test set is sampled with additional M test images of each of the N classes, such that $I^{test} = \{I_1^1, \ldots, I_N^K; y_1^1, \ldots, y_N^K\}$. The test set is also referred to as the query set. For meta-learning a collection of G meta-training tasks as $T = \{(t_i^{train}, t_i^{test})\}_{g=1}^G$ is sampled, where each tuple $(t_i^{train}, t_i^{test})$ describes the support and query sets of the corresponding meta-task. After meta-training, the performance of the model is evaluated on meta-testing set $S = \{(t_i^{train}, t_i^{test})\}_{g=1}^G$ for a different set of object classes with the same numbers of samples in the support and query sets. In this paper, we train our model on the combined meta-training set $T$ through image classification. After our model is trained, we directly test on meta-testing set $S$ without fine-tuning the model parameters.

3.2. Learning a part dictionary via clustering

Formulation. We denote a feature map $F^l \in \mathbb{R}^{H \times W \times C}$ as the output of a layer $l$ in a deep convolutional neural network, with $C$ being the number of channels. A feature vector $f^l_p \in \mathbb{R}^C$ is the vector of features in $F^l$ at position $P$ on the 2D lattice $\mathcal{P}$ of the feature map. In the remainder of this section, we omit the superscript $l$ for notational clarity because this is fixed prior.

Learning semantic parts. A number of prior works [16, 39, 13, 12] on learning compositional representations showed that when clustering feature vectors $f_p$, the cluster centers resemble image patterns that frequently re-occur in the training images. These patterns often share semantic meanings and therefore resemble part-like detectors. Motivated by these results, we aim at constructing a dictionary $D = \{d_1, \ldots, d_B\}$, with components $d_b \in \mathbb{R}^C$ part representations in the intermediate layer $F$ of a DCNN. We initialize the part dictionary $D$ via K-means clustering on the feature vectors $f_p$. Each cluster center can intuitively be considered to capture the mid-level semantics. The part dictionary $D$ will be updated at the meta-training stage. For better supervision on it, we add an additional cluster loss to the loss function when training the network, which will be introduced in section 3.4. Figure 3 illustrates examples of the dictionary components $d_b$ after the meta-learning stage by showing image patches that activate each component the
most. Note how part representations indeed capture semantically meaningful image patterns, such as the head of a dog.

3.3. Compositional Part Sharing for Few-Shot Classification

Computing the spatial activation patterns of parts. Given the part dictionary \( D \), we compute the activation of a part representation \( d_b \) at a position \( p \) in the feature map \( F \) by computing the cosine similarity between \( d_b \) and the feature vector \( f_p \). We implement this module as a convolution layer, which we call part detection layer. The convolutional kernels of this part detection layer are the components of the part dictionary \( D \), and their kernel size is \( 1 \times 1 \). The output of part the detection layer is a part activation tensor \( A \) \( \in \mathbb{R}^{H \times W \times B} \), where \( B \) is the number of components in dictionary \( D \). Each channel in this tensor \( A_b \) \( \in \mathbb{R}^{H \times W} \) is referred to as part activation map.

Learning dictionaries of spatial activation patterns. Our goal is to enable the model to share part activation patterns among different classes. This is inspired by the idea that parts of different objects can have similar spatial activation patterns and that this natural redundancy should be exploited. We achieve this by learning a dictionary of part activation patterns \( S = \{S_1, ..., S_V\} \), namely map dictionary, which contains the most part activation patterns in the training data. We integrate the dictionary components \( S_v \in \mathbb{R}^{H \times W} \) into the feed-forward stage by comparing them to the individual part activation maps \( A_b \) using the cosine similarity. We then select the closest component \( \hat{v} = \arg\min_v \cos(S_v, A_b) \) and compute the output channel as point-wise multiplication between \( A_b \) and \( S_{\hat{v}} \). After repeating this operation for all spatial distribution maps, we get the activated spatial distribution output notated as \( O \in \mathbb{R}^{H \times W \times B} \). In this way, each part activation map \( A_b \) is encouraged to learn information from the most similar stored spatial activation pattern \( S_v \).

Augmenting important parts with attention. To further augment parts that are most important for representing a particular object, we adopt an attention mechanism to calculate different weights and the relationship between the parts’ spatial distributions. We follow the design of SENet [10] with small changes. In particular, we first squeeze the global spatial information of \( O \) into a channel descriptor by using a convolutional filter. Formally, a summary vector \( z \in \mathbb{R}^C \) is generated by shrinking \( O \) through its spatial dimensions \( H \times W \), such that the \( b \)-th entry of the vector \( z_b \) is calculated by:

\[
z_b = r \otimes O_b \in \mathbb{R}
\]

where \( r \) refers to convolution kernel. To fully exploit the squeezed information, we then use the same gating mechanism as SENet which contains a bottleneck with two fully-connected layers and non-linearity activation. It can be represented as

\[
l = \sigma(W_2 \delta(W_1 z)) \in \mathbb{R}^B
\]

where \( \sigma \) refers to the Sigmoid activation and \( W_1, W_2 \) are the weights of the fully-connected layers. With the computed activation \( l \), the final output is obtained by re-weighting the input \( O \) with \( l \):

\[
\Phi_b = l_b \cdot O_b
\]

where \( \cdot \) refers to channel-wise multiplication between the scalar \( l_b \) and the channel output \( O_b \). Finally, we normalize feature vectors along channel dimension in \( \Phi \) to have unit norm and concat it with average-pooled \( F \) then forward it into the classifier to obtain a final prediction.
3.4. End-to-end Training of the model

During training, we use a two-layer fully-connected structure as a classifier to predict the classification results. Our model is fully differentiable and can be trained end-to-end using back-propagation. The trainable parameters of our model are $\Theta = \{\Omega, D, S\}$, where $\Omega$ are the parameters of the backbone used for feature extraction, e.g., ResNet-12. $D$ is the part dictionary, and $S$ is the dictionary of part activation maps. We optimize these parameters jointly using stochastic gradient descent. Our loss function contains three terms:

$$
L(y, y') = L_{\text{class}}(y, y') + \gamma_1 L_{\text{cluster}}(D) + \gamma_2 L_{\text{sparse}}(S) \tag{4}
$$

$L_{\text{class}}(y, y')$ is the cross-entropy loss between the predicted label $y'$ and the ground-truth label $y$. The second term $L_{\text{cluster}}(D)$ is used to add additional regularization for the dictionary of parts:

$$
L_{\text{cluster}}(D) = \sum_p \min_b (1 - \cos(D_b[f_p])) \tag{6}
$$

where $f_p$ refers to the feature vector at position $p$ in the feature map $F$ and $\cos(\cdot, \cdot)$ refers to the cosine similarity. Intuitively, this loss encourages the dictionary’s components to become similar to the feature vectors $f_p$. Thus the dictionary is forced to learn part representations that frequently occur in the training data.

To regularize the map dictionary, we add a sparse loss on the dictionary $S$:

$$
L_{\text{sparse}} = \sum_{v=1}^V \max_{v'} \tau(S_v, S_{v'}) \tag{7}
$$

where $\tau(S_v, S_{v'})$ measures the similarity between two dictionary elements of $S$. The similarity is split into two cases according to a fixed threshold $\alpha$:

$$
\tau(S_v, S_{v'}) = \begin{cases}
\cos(S_v, S_{v'}), & \text{if } \cos(S_v, S_{v'}) > \alpha \\
0.5 \times \cos(S_v, S_{v'})^2, & \text{if } \cos(S_v, S_{v'}) \leq \alpha
\end{cases} \tag{8}
$$

This regularizer encourages the map dictionary elements to be sparse, thus avoiding that the elements become too similar to each other. We find that splitting the term into two situations based on the threshold $\alpha$ helps the model converge during training.

3.5. Replacing the classification head during metatesting

In the meta-testing stage, what differs us from many other existing models is that we do not further fine-tune our model based on the support sets $D_j^{\text{train}}$. Instead, we replace the fully-connected classification head with a linear classifier head since training a new fully-connected layer on the support set will lead to overfitting. We tried nearest neighbors based on cosine distance, and several linear classifiers such as logistic regression, linear support vector machine. We found that the logistic regression gives the best results. To summarize, for a task $(D_j^{\text{train}}, D_j^{\text{test}})$ sampled from meta-testing set $S$, we forward $D_j^{\text{train}}$ through the whole embedding function to get the attentioned part activation map $F$ contacted with average-pooled $F$, and train the logistic regression classifier on this representation.

4. Experiment

In this section, we conduct extensive experiments that prove the effectiveness of our model. We first describe our detailed setup in section 4.1, which includes datasets, model structure, and hyper-parameters. Then we evaluate our model and make a comparison to related work on four few-shot classification benchmark datasets: miniImageNet [33], tieredImageNet [26], CIFAR-FS [1], Fewshot-CIFAR100 (FC100) [21]. The concrete performance on ImageNet derivatives is introduced in section 4.2 and that on CIFAR derivatives is introduced in section 4.3. We further conduct ablation studies in section 4.4 to demonstrate the effects of the individual modules of our COMPAS pipeline.

4.1. Experimental Setups

Architecture. Following previous work [18, 21, 15, 25, 3], we use a ResNet12 as our feature extraction network which contains 4 residual blocks, where each of them contains 3 convolution layers. The filter numbers of the last convolution layer in each block are changed from (64,128,256,512) to (64,160,320,640). We make another small modification in that we drop the last average-pooling layer. Dropblock is used in our model as a regularizer. We
set the number of components in the part dictionary \( D \) to 512 and the number of components in the map dictionary \( S \) to 2048 in our experiments.

**Implementation details.** We use the SGD optimizer with a momentum of 0.9 and a weight decay of \( 5 \times 10^{-4} \). Our batch size is set to 64, and the base learning rate is 0.05. We report the performance on testing sets with embedding functions trained on the combination of training sets and validation sets. On miniImageNet and tieredImageNet, we train our model 100 epochs and for CIFAR derivatives, the total epochs for training are 90. We adopt cosine annealing as the learning rate scheduler. During training, we adopt regular data augmentation schemes such as random flipping, color jittering. When handling CIFAR derivatives datasets, we resize the input image to \( 84 \times 84 \) pixels in order to have enough spatial resolution. Following common experimental setups, we report our performance based on an average of 600 meta-tasks, where each of them contains 15 test instances per class.

### 4.2. Experiments on ImageNet derivatives

The miniImageNet dataset is the most classic few-shot classification benchmark proposed by Matching Networks [33]. It consists of 100 randomly sampled different classes, and each class contains 600 images of size \( 84 \times 84 \) pixels. We follow the widely-used splitting protocol proposed by Ravi et al. [24], which uses 64 classes for meta-training, 16 classes for meta-validation, and 20 classes for meta-testing.

The tieredImageNet dataset is a larger subset of ImageNet, composed of 608 classes grouped into 34 high-level categories. They are further divided into 20 categories for meta-training, 6 categories for meta-validation, and 8 categories for meta-testing, which corresponds to 351, 97, and 160 classes for meta-training, meta-validation, and meta-testing, respectively. This splitting method, which considers high-level categories, is applied to minimize the semantic between the splits. Images are of size \( 84 \times 84 \).

**Results.** Table 1 summarizes the results on the 5-way miniImageNet and tieredImageNet. Our method achieves state-of-the-art performance on the miniImageNet benchmark for both 5-way-1-shot and 5-way-5-shot tasks. On tieredImageNet, we also achieve the best performance on the 5-way-5-shot task and comparable performance on the 5-way-1-shot task. Note that LEO [27] used an encoder and relation network in addition to the WRN-28-10 backbone network to produce sample-depend initialization of the gradient descent. FEAT [36] and LEO [27] pre-train the WRN-28-10 backbone to classify 64 meta-training set of miniImageNet and then continue meta-training. TADAM [21] co-trained the feature embedding on both meta-training task (5-way) and the standard classification task (64-way) together. FEAT [36] and MABAS [11] require additional fine-tuning on meta-testing sets. By contrast to all those approaches, our model just needs to train the embedding function through standard classification without further fine-tuning. This strategy allows us to clearly demonstrate the effect of a good embedding function by achieving stronger performance with an arguably simpler training. We also no-
Table 2. Comparison to prior work on CIFAR-FS and FC100. Average few-classification accuracies(%) with 95% confidence intervals on the meta-testing sets of CIFAR-FS and FC100. a-b-c-d denotes a 4-layer convolutional network with a, b, c, d filters in each layer.

| model               | backbone   | CIFAR-FS 5-way |          | FC100 5-way |          |
|---------------------|------------|----------------|----------|-------------|----------|
|                     |            | 1-shot         | 5-shot   | 1-shot      | 5-shot   |
| MAML [5]            | 32-32-32-32| 58.9 ± 1.9     | 71.5 ± 1.0| -           | -        |
| Prototypical Networks [29] | 64-64-64-64 | 55.5 ± 0.7     | 72.0 ± 0.6| 35.3 ± 0.6  | 48.6 ± 0.6|
| Relation Networks [31] | 64-96-128-256 | 55.0 ± 1.0     | 69.3 ± 0.8| -           | -        |
| R2D2 [1]            | 96-192-384-512 | 65.3 ± 0.2   | 79.4 ± 0.1| -           | -        |
| TADAM [21]          | ResNet-12  | -              | -        | 40.1 ± 0.4  | 56.1 ± 0.4|
| Shot-Free [25]      | ResNet-12  | 69.2 ± n/a     | 84.7 ± n/a| -           | -        |
| TEWAM [22]          | ResNet-12  | 70.4 ± n/a     | 81.3 ± n/a| -           | -        |
| Prototypical Networks [29] | ResNet-12 | 72.2 ± 0.7     | 83.5 ± 0.5| 37.5 ± 0.6  | 52.5 ± 0.6|
| MetaOptNet [15]     | ResNet-12  | 72.6 ± 0.7     | 84.3 ± 0.5| 41.1 ± 0.6  | 55.5 ± 0.6|
| DeepEMD [37]        | ResNet-12  | -              | -        | 46.47 ± 0.78| 63.22 ± 0.71|
| RFS [32]            | ResNet-12  | 73.9 ± 0.8     | 86.9 ± 0.5| 44.6 ± 0.7  | 60.9 ± 0.6|
| MABAS [11]          | ResNet-12  | 73.51 ± 0.96   | 85.49 ± 0.68| 42.31 ± 0.75| 57.56 ± 0.78|
| Ours                | ResNet-12  | 74.21 ± 0.68   | 87.68 ± 0.47| 46.72 ± 0.66| 62.84 ± 0.72|

Table 3. Ablation study. Performance of our ablated models on four few-shot classification benchmarks. The metric is average few-classification accuracies(%).

| Attention | Cluster Loss | Sparse Loss | miniImageNet 1-shot | tieredImageNet 1-shot | CIFAR-FS 5-way 1-shot | FC100 5-way 1-shot | miniImageNet 5-shot | tieredImageNet 5-shot | CIFAR-FS 5-way 5-shot | FC100 5-way 5-shot |
|-----------|--------------|-------------|----------------------|-----------------------|-----------------------|---------------------|----------------------|-----------------------|-----------------------|---------------------|
| ✓         |              |             | 60.58                | 78.24                 | 70.52                 | 82.44               | 70.47                 | 83.60                 | 41.77                 | 58.68               |
| ✓         | ✓            |             | 61.79                | 79.88                 | 71.44                 | 83.90               | 72.15                 | 84.97                 | 42.67                 | 59.85               |
| ✓         | ✓            | ✓           | 64.81                | 82.05                 | 72.90                 | 86.49               | 73.50                 | 86.07                 | 45.69                 | 61.12               |
| ✓         | ✓            | ✓           | 66.37                | 83.44                 | 74.15                 | 86.96               | 74.21                 | 87.68                 | 46.72                 | 62.84               |

4.3. Experiments on CIFAR derivatives

The CIFAR-FS dataset is a recently proposed few-shot image classification benchmark derived from CIFAR. It consists of all 100 classes and is further randomly split into 64 meta-training classes, 16 meta-validation classes, and 20 meta-testing classes. Each class contains 600 images of size 32 × 32.

The FC100 dataset is another few-shot classification dataset based on CIFAR. Its main idea is very similar to tieredImageNet, where the whole 100 classes are grouped into 20 superclasses. Each superclass is composed of standard 5 classes. These superclasses are divided into 12, 4, 4 for meta-training, meta-validation, meta-testing correspondingly. Images are of size 32 × 32 pixels.

Results. Table 2 summarizes the performance on the 5-way CIFAR-FS and FC100. Our model achieves state-of-the-art performance on the CIFAR-FS benchmark and the 5-way-1-shot task on the FC100 benchmark. We also achieve comparable results on the 5-way-5-shot task on FC100. We observe that the relative improvement rate on the CIFAR-FS dataset is larger compared to the FC100 dataset. We have observed a similar generalization pattern on the ImageNet derivatives. Namely, the performance on the benchmark with semantic gaps between the meta-training set and meta-testing set benefits less from our method.

4.4. Ablation Experiments

In this section, we conduct ablation studies on our COMPAS pipeline to analyze how its variants affect the few-shot classification result. We study the following three components of our method: (a) The attention module on activated spatial distribution maps; (b) The cluster loss of the part dictionary; (c) The sparse loss of the map dictionary. In addition, we also analyze the result of the number of components in the map dictionary $S$ and how the different formations of meta-training and meta-testing sets affect the final result.

Table 3 shows the result of our ablation studies on miniImageNet, tieredImageNet, CIFAR-FS and FC100. We can see that the attention mechanism for augmenting important parts and their relationship makes the average performance improve around 1.5% on all datasets. With our cluster loss that regularizes the components in the part dictionary $D$, we gain 2.5%. In addition, this loss increases the interpretability of our model as it makes the image patches detected by these part detectors more semantically meaningful. Our sparse loss regularizer improves the performance by another
Table 4. **Reorganization of tieredImageNet and FC100.** We compare the performance of our model on the original tieredImageNet and FC100 with the reorganized tieredImageNet and FC100 (see text). The metric is average few-classification accuracies(%) . O stands for the original split, R stands for the reorganized split.

| model | backbone | tieredImageNet-O 1-shot | tieredImageNet-R 1-shot | tieredImageNet-O 5-shot | tieredImageNet-R 5-shot | FC100-O 1-shot | FC100-R 1-shot | FC100-O 5-shot | FC100-R 5-shot |
|-------|----------|-------------------------|-------------------------|-------------------------|-------------------------|----------------|----------------|----------------|----------------|
| Ours  | ResNet-12 | 74.15                   | 86.96                   | 76.30                   | 88.05                   | 46.72          | 62.84          | 49.65          | 66.70          |

Figure 4. **t-SNE visualization illustrating improved feature embeddings with our designed modules.** The left figure corresponds to our model without attention module, cluster loss, and sparse loss. The right figure corresponds to the complete model.

Figure 5. **Test accuracies(%) on meta-testing sets with a varying number of components in the map dictionary.** The performance of our model increases at first and saturates at some point with a slight tendency to drop for large numbers of components.

1%, which demonstrates the benefit of making the components in the map dictionary distinct from each other.

In Figure 4, we further show the t-SNE visualization results of the embedding space of our model on the CIFAR-FS dataset. For both t-SNE plots, we use the same data and the same hyper-parameters. The left figure shows the embedding space without using attention, the cluster, and sparse loss for regularization. The right figure shows the result with the full model. We can observe that even without these modules, the visualization result is quite good compared to previous work. However, after adding these modules, the cluster centers are forced to become even denser and more distinct from each other.

In Table 4, we validate the hypothesis that our model works better when the meta-testing sets and the meta-training sets share similar classes. To reorganize the two datasets, we randomly sample classes from the original meta-training sets and meta-testing sets to form a new partition. In this way, the classes within the superclasses are split into meta-training sets and meta-testing sets such that they will have a semantic overlap. The results show that by reorganizing these two datasets, we get around 2.5% improvement on the 1-shot task and 3% improvement on the 5-shot task. This happens because, in this situation, we can share more knowledge during the meta-testing stage from both the part dictionary and the map dictionary learned during the meta-training stage.

Figure 5 illustrates the influence of the number of components in the map dictionary $S$ on the performance of our model on four benchmarks. The performance improves at first when the number of components increases but saturates as the dictionaries become larger. The performance keeps at the same level and even shows a tendency to drop. These results suggest that when the capacity of the dictionary is small, our model cannot store all necessary information. However, if the capacity becomes too large, the model starts to overfit slightly.

5. **Conclusion**

In this work, we study the problem of few-shot image classification. Inspired by the compositional representation of objects in humans, we introduce COMPAS, a novel neural architecture for few-shot classification that learns through compositional part sharing. In particular, COMPAS learns a knowledge base that contains a dictionary of part representations and a dictionary of part activation maps that encode frequent spatial activation patterns of parts. During meta-testing, this knowledge is reused to recognize unseen classes from very few samples. Our extensive experiments demonstrate the effectiveness of our method by achieving state-of-the-art performance on four popular few-shot classification benchmarks.
References

[1] Luca Bertinetto, João F. Henriques, Philip H. S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers, 2019. 2, 5, 6, 7
[2] Jifeng Dai, Yi Hong, Wenzhe Hu, Song-Chun Zhu, and Ying Nian Wu. Unsupervised learning of dictionaries of hierarchical compositional models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014. 3
[3] Gunnet S. Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification, 2020. 5
[4] Sanja Fidler, Marko Boben, and Ales Leonardis. Learning a hierarchical compositional shape vocabulary for multi-class object representation, 2014. 3
[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. 1, 2, 6, 7
[6] Mohammad Ghasedzadeh, Fang Lin, Bita Darvish Rouhani, Farinaz Koushanfar, and Ke Huang. Agilenet: Lightweight dictionary-based few-shot learning, 2018. 1
[7] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 1, 2, 6
[8] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Oct 2017. 1
[9] 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. 1, 2
[10] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 1, 2, 6
[11] Jaekyeom Kim, Hyoungseok Kim, and Gunhee Kim. Model-agnostic boundary-adversarial sampling for test-time generalization in few-shot learning. 6, 7
[12] Adam Kortylewski, Ju He, Qing Liu, and Alan L. Yuille. Compositional convolutional neural networks: A deep architecture with innate robustness to partial occlusion. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 3
[13] Adam Kortylewski, Qing Liu, Huiyu Wang, Zhishuai Zhang, and Alan Yuille. Combining compositional models and deep networks for robust object classification under occlusion. arXiv preprint arXiv:1905.11826, 2019. 3
[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012. 1
[15] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019. 1, 2, 5, 6, 7
[16] Renjie Liao, Alex Schwing, Richard Zemel, and Raquel Urtasun. Learning deep parsimonious representations. In Advances in Neural Information Processing Systems, pages 5076–5084, 2016. 2, 3
[17] Bin Liu, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Menglong Zhu, and Han Hu. Negative margin matters: Understanding margin in few-shot classification, 2020. 6
[18] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner, 2018. 5
[19] Tsendure Munkhdalai, Xingdi Yuan, Soroush Mehr, and Adam Trischler. Rapid adaptation with conditionally shifted neurons. volume 80 of Proceedings of Machine Learning Research, pages 3664–3673, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. 6
[20] Alex Nichol and John Schulman. Reptile: a scalable meta-learning algorithm. arXiv: Learning, 2018. 1, 2
[21] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 31, pages 721–731. Curran Associates, Inc., 2018. 1, 2, 5, 6, 7
[22] Limeng Qiao, Yemin Shi, Jia Li, Yapowei Wang, Tiejun Huang, and Yonghong Tian. Transductive episodic-wise adaptive metric for few-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019. 6, 7
[23] Siyuan Qiao, Chenxi Liu, Wei Shen, and Alan L. Yuille. Few-shot image recognition by predicting parameters from activations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 1, 2
[24] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016. 6
[25] Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Few-shot learning with embedded class models and shot-free meta training. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019. 5, 6, 7
[26] 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, 2018. 2, 5
[27] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization, 2019. 6
[28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. 1
[29] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30, pages 4077–4087. Curran Associates, Inc., 2017. 1, 2, 6, 7
[30] Yihong Sun, Adam Kortylewski, and Alan Yuille. Weakly-supervised amodal instance segmentation with compositional priors, 2020. 3

[31] 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 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 1, 2, 6, 7

[32] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need?, 2020. 1, 2, 6, 7

[33] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 29, pages 3630–3638. Curran Associates, Inc., 2016. 1, 2, 5, 6

[34] Fangyu Wu, Jeremy S. Smith, Wenjin Lu, Chaoyi Pang, and Bailing Zhang. Attentive prototype few-shot learning with capsule network-based embedding. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, Computer Vision – ECCV 2020, pages 237–253, Cham, 2020. Springer International Publishing. 3

[35] Ya Jin and S. Geman. Context and hierarchy in a probabilistic image model. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06), volume 2, pages 2145–2152, 2006. 3

[36] 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/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 1, 2, 6

[37] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover’s distance and structured classifiers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2020. 6, 7

[38] Jian Zhang, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaoang Yang. Variational few-shot learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October 2019. 6

[39] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018. 2, 3

[40] Zhishuai Zhang, Cihang Xie, Jianyu Wang, Lingxi Xie, and Alan L Yuille. Deepvoting: A robust and explainable deep network for semantic part detection under partial occlusion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1372–1380, 2018. 2

[41] Long (Leo) Zhu, Chenxi Lin, Haoda Huang, Yuanhao Chen, and Alan Yuille. Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, Computer Vision – ECCV 2008, pages 759–773, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 3