Canonical mean filter for almost zero-shot multi-task classification

Yong Li · Heng Wang · Xiang Ye

Abstract

The support set plays a key role in providing conditional prior for fast adapting the feature extractors in few-shot tasks. The representative few-shot method CNAPs used a simple conditional feature encoder to extract the prior, which was a conditional feature of the task-specific support set. However, the strict form required by the support set makes its construction difficult in practical applications. Motivated by ANIL, we first investigate the role of the adaption in CNAPs by designing Almost Zero-Shot (AZS) tasks in which a fixed support set is used across all tasks instead of the task-specific support set. The AZS experimental results indicate that the adaptation contributes little to the feature extractor. Nevertheless, simply removing the adaptation from CNAPs will cause severe performance drop on some sub-datasets of the Meta-Dataset. To alleviate this problem, we further proposed a Canonical Mean Filter (CMF) module that maps any support set to a canonical form. The CMF yields a stable conditional feature, and hence allows for CNAPs to perform well without the conditional feature encoder and the parameter adaptation network at the test stage. CNAPs embedded with CMF significantly outperform the vanilla CNAPs for the AZS tasks, and even achieve comparable performance to one-shot tasks but with 40.48% parameter reduction and without task-specific support sets at the test stage.

Keywords Adaptation · Feature reuse · Support set

1 Introduction

A critical step in few-shot learning is to conduct the adaptation with the prior information provided by the support sets [1]. Ideally, a support set should contain some samples with corresponding labels for each category [1]. Existing works often construct a support set for every particular task which deals with only a limited and known number of categories. In real few-shot applications, however, it is almost impossible to construct a proper support set for the whole test dataset due to the unknown number of object categories and their labels in it. Since the prior information provided by an improper support set may misguide the adaptation, this work examines the role of feature adaptation and the generalization ability of the trained model when proper support sets are unavailable.

In literature, meta-learning methods mainly include two categories, the optimization-based and the metric-based methods. For optimization-based Model-agnostic Meta-Learning [2] (MAML), Almost No Inner Loop [3] (ANIL) is proposed to investigate the role of the adaptation of the feature extractor by stopping the inner loop of MAML at the test stage. ANIL finds that the feature adaptation contributed much less than the feature reuse in the generalization ability of MAML-based models. Motivated by ANIL, this work studies the role of the adaptation of the feature extractors in CNP-based few-shot methods. The particular experiments are conducted on Conditional Neural Adaptive Process (CNAPs) [4], a representative CNP-based few-shot method. CNAPs are evaluated on a large-scale and complex dataset, Meta-Dataset [5], containing ten cross-domain common datasets while other CNPs-based methods mainly focus on in-domain transferring ability. Since the good cross-domain transferring ability of CNAPs, this work first studied the contribution of feature adaptation and feature reuse in CNAPs and then generalized the study to other CNP-based methods.
Fig. 1 The clustering results of the extracted conditional features on Omniglot test images. The red and cyan points are the clustering results of the conditional features yielded by CNAPs-CMF and CNAPs, respectively.

We first analyze CNAPs, consisting of four parts: a conditional feature encoder, an adaptation network, a ResNet-18 pre-trained on the ImageNet-1K (as the inference model), and a linear classifier. The support samples are encoded into the conditional features for each task by the conditional feature encoder. Then the conditional features are input into the adaptation network to yield the transformation parameters, which are used to adapt the parameters of the pre-trained ResNet-18. The adapted ResNet-18 extracts the features of target samples, and then the features are classified by the linear classifier.

Then this paper designs two ‘Almost Zero-Shot’ (AZS) experiments, AZS-I and AZS-II, to analyze the role of feature adaptation in CNAPs from the in-domain and cross-domain aspects. In AZS-I, we randomly choose a support set from each sub-dataset to replace all support sets in all tasks of this sub-dataset. The chosen support set freezes the conditional features; hence the subsequent transformation parameters are fixed. This stops the adaptation of CNAPs on each sub-dataset, i.e., in-domain adaptation is stopped, and only a slight performance drop is observed.

In AZS-II, a single random but fixed support set is chosen from a particular sub-dataset. The fixed support set is used to replace all support sets across all sub-datasets to stop all the adaptation for CNAPs, i.e., the cross-domain adaptation is stopped. Again, only a slight performance drop is observed on seven sub-datasets (see Table 3) when the fixed support set is randomly selected from ImageNet. This observation coincides with ANIL [3] in that task-specific support sets still necessary for multi-dataset few-shot learning methods. However, a dramatic performance drop was observed on a few sub-datasets: Omniglot [6], Quick Draw, and Aircraft sub-datasets. Thus, a question arises: are the task-specific support sets still necessary for multi-dataset few-shot learning, or does the feature extractor of CNAPs rely too much on the support sets?

By deeply analyzing CNAPs, we find that the feature extractor is over-sensitive to the input support set, i.e., a small change in the support set may cause a dramatic adaptation shift when guiding the pre-trained ResNet-18. The scattered cyan points in Fig. 1 show an example of conditional features of corresponding support sets for different tasks. Obviously, the conditional features change a lot when the support samples slightly change. This requests that the conditional encoder is robust enough and the support samples are elaborately selected to be close enough to the corresponding target samples in each task. A fixed support set randomly chosen from a particular sub-dataset cannot satisfy the close request. Hence in AZS-II tasks, CNAPs performs poorly across sub-datasets.

To address the over-sensitivity problem, we proposed a dynamic-kernel-like module, Canonical Mean Filter (CMF), which is a lightweight module that can be embedded in the conditional feature encoder of vanilla CNAPs (called CNAPs-CMF). CMF first applies a designed attention module to generate a vector for each support sample and then fuses the attention vectors over all support samples. Then the fused vector serving as the set of weights is assigned to the kernels in the next layer. The CMF-embedded kernels map varying samples to a canonical form and become less sensitive to the input samples, so the conditional features extracted by CMF are stable even when the support sets are randomly generated. The red points in Fig. 1(a) and (b) show the intensive and stable conditional features extracted by CMF.

Similar to ANIL [3], the proposed CMF provides a better generalization (feature reuse). To investigate the contribution of feature reuse to classification performance, we conducted various experiments on both AZS-II and one-shot tasks.

---

1 We viewed a single sub-dataset as a domain. Hence the adaptation within a sub-dataset is called in-domain adaptation, and the adaptation between the sub-datasets is called cross-domain adaptation.
Since the classifiers require the support sets [3], which means the adaptation still contributes to the classification performance, to separate the feature reuse from the adaptation, we resort to analyzing the separability of the features extracted by CNAPs-CMF. The analysis shows that they have a much smaller inner-class distance than CNAPs-extracted features, while maintaining a comparable inter-class distance (see detail in Sect. 4.2), indicating better separability.

As a result, the feature adaptation can be stopped in CNAPs-CMF, i.e., the adaptation parameters can be pre-computed and stored before the test stage to avoid repetitive computation. Then the stored adaptation parameters can be reused at the test stage. So the conditional feature encoder and parameter adaptation networks are removed at the test stage, reducing the parameter amount by 40.48%. The main contributions are concluded as follows.

1. Motivated by ANIL, we show that the adaptation is less important in CNP-based few-shot methods by the designed Almost Zero-Shot (AZS) tasks, AZS-I and AZS-II. In AZS-II tasks, the adaptation can be completely stopped.
2. We proposed CMF to address the over-sensitivity problem of the conditional feature encoder by mapping the varying support sets to a canonical form. The canonical form means stable conditional features, enabling CNAPs-CMF to outperform CNAPs by a large margin on AZS-II and one-shot tasks. The stable conditional features help remove the construction of the support sets.
3. After CNAPs-CMF is fully trained, the adaptation of the feature extractor is stopped. Naturally, the structures responsible for feature adaptation can be removed at the test stage, and only the frozen ResNet-18 and the stored adaptation network of linear classifiers need to be stored. This reduces the number of parameters by 40.48%.

The rest of the paper is organized as follows. Section 2 overviews the meta-learning and few-shot learning methods; Sect. 3 formulates CNAPs and CNPs [7] in a mathematical form, then presents the canonical mean filter (CMF); Sect. 4 presents the experimental results on multi-task classification, and conducts the ablation experiments to analyze the proposed CMF; and the conclusion is given in Sect. 5.

2 Related works

Meta-learning-based few-shot learning methods use the prior provided by support sets to transfer the target features. These methods can be divided into metric-based, gradient-based, and CNPs-based. Metric-based methods compute class-wise prototypical vectors from support vectors and classify target vectors based on their distances to the prototypical vectors. Gradient-based methods train a support sample encoder to extract gradients, transferring the target sample encoder. As a representative, MAML consists of outer and inner loops, representing target and support set encoders. CNPs-based methods aggregate support vectors to form a task-specific prior vector, transferring the target sample encoder. CNPs use neural networks to mimic stochastic processes and develop task-dependent priors to transfer the target set encoder. NPs [8], ANPs [9], ConvCNPs [10], and CNAPs [4] are variations of CNPs for 1-D signal recognition [11–13].

Among the three categories, gradient-based and CNPs-based methods explicitly utilize the prior encoded by the support set encoder to transfer target features. ANIL demonstrated the effectiveness of the transferring module in gradients by stopping the explicit transfer in MAML, showing that the parameter transfer contributes little to the performance. To explore the effectiveness of support sets, we stopped the explicit transfer in CNAPs, a representative CNPs-based method.

AZS-I and AZS-II experiments demonstrate that the support set encoder is over-sensitive to the input support set. This problem is caused by maximum likelihood optimization, encouraging the whole model to fit the task-specific pattern and leads to scattered prior features. To address this problem, we propose using the marginal likelihood optimization, encouraging the model to fit the mean pattern of all tasks and hence generating stable prior features.

Patacchiola et al. proposed DKT [14] to realize marginal likelihood optimization by approximating the kernel parameterized by $\theta$ with the kernel of the Gaussian Process. However, GP kernels do not well approximate a wide variety of unknown kernels. The proposed CMF uses Dynamic convolution (DY) [15] technology to estimate stable prior features without directly estimating the kernel. Unlike DY, which generates sample-dependent features for classification tasks, CMF aims to generate stable sample-independent features across different tasks.

3 Proposed method

This section presents the canonical mean filter (CMF) that is able to map varying support sets to a canonical form. To clearly state the rationale of CMF, we first need to formulate CNAPs mathematically. Based on the formulation, the conditional feature encoder is analyzed and shown to be over-sensitive to the varying support sets. Then we propose CMF to alleviate the over-sensitiveness by mapping the support samples into a canonical form, and provide a theoretical explanation from the Bayesian inference view.
3.1 Mathematical formulation of CNAPs

CNAPs originate from CNPs. For completeness, CNPs [7] are briefly introduced here. Let \( \mathcal{X}^C \) and \( \mathcal{X}^T \) denote the support sample dataset and target image dataset, and \( \gamma^C \) and \( \gamma^T \) denote their labels.

CNPs use Neural Networks (NNs) to build the conditional feature encoder parameterized by \( \theta \) from the support sample dataset to the feature space, \( F(\cdot, \theta) : \mathcal{X}^C \rightarrow \mathcal{R}^C \). \( \mathcal{R}^C = F(\mathcal{X}^C, \theta) \) encodes the conditional information of support sets \( \{\mathcal{X}^C, \gamma^C\} \) and is used to generate the adaptation parameter \( \phi \) for the inference model by

\[
\pi(\cdot, \mathcal{R}^C) : \varphi \rightarrow \phi,
\]

where \( \varphi \) represents the original parameter of the inference model. We call the parameter adaptation mean shifting in this work. Then, the inference model \( G(\phi, \cdot) \) predicts the classes of the target set \( \mathcal{X}^T \) by

\[
y_{\text{pred}}^{\pi} = G\left(\phi, \mathcal{X}^T\right) = G\left(\pi\left(\varphi, F(\mathcal{X}^C, \theta)\right), \mathcal{X}^T\right).
\]

(1)

In the training process, CNPs optimize \( G \) jointly over \( \theta \) and \( \varphi \) in Eq. 1 by maximizing the likelihood given \( (\mathcal{X}^T, y^T) \). Formally,

\[
\hat{\theta}, \hat{\varphi} = \arg\max_{\theta, \varphi} p\left(y^T | G\left(\pi\left(\varphi, F(\mathcal{X}^C, \theta)\right), \mathcal{X}^T\right)\right).
\]

(2)

Following CNPs, CNAPs [4] build a simple convolutional neural network to yield \( \mathcal{R}^C \), then use a mean operator to aggregate the features over all samples to get the mean vector \( \mathcal{R}_m^C \):

\[
\mathcal{R}_m^C = \frac{1}{S} \sum_{i=1}^{S} \mathcal{R}_i^C,
\]

where \( S \) is the cardinality of the input support sets.

With FiLM layers [16], CNAPs design an adaptation network parameterized by \( \gamma = \{\gamma_w, \gamma_b\} \) to realize the mapping \( \pi \) in Eq. 1. Formally,

\[
\hat{\phi} = \pi(\varphi, \gamma, \mathcal{R}_m^C) = (\gamma_w \otimes \mathcal{R}_m^C) \times \varphi + (\gamma_b \otimes \mathcal{R}_m^C),
\]

(3)

where \( \otimes \) represents the convolution operation.

To build a powerful feature extractor for the inference model, CNAPs use the ResNet-18 [17] with every convolutional layer being followed by FiLM layers [16]. ResNet-18 is pre-trained on ImageNet-1K and outputs 512-D features. The parameters \( \varphi \) of the ResNet18 are frozen, i.e., they will not be updated in the subsequent training process of CNAPs. \( \pi(\varphi, \gamma, \mathcal{R}_m^C) \) adapts \( \varphi \) of the frozen fully-trained ResNet-18 model to a new target dataset according to \( \mathcal{R}_m^C \) and \( \gamma \). With \( \pi \), the maximum likelihood in Eq. 2 will be optimized as

\[
\hat{\theta}, \hat{\gamma} = \arg\max_{\theta, \gamma} p\left(y^T | G\left(\pi\left(\varphi, \gamma, F(\mathcal{X}^C, \theta)\right), \mathcal{X}^T\right)\right).
\]

(4)

In Eq. 4, CNAPs optimize \( G(\varphi, \mathcal{X}^T) \) jointly over \( \theta \) and \( \gamma \). Since \( \gamma \) is also driven by \( \mathcal{R}_m^C \), i.e., the over-sensitivity comes from \( F(\cdot, \theta) \). Designing a good conditional feature encoder \( F(\cdot, \theta) \) is a critical step to alleviate the over-sensitivity of CNAPs. This helps \( \mathcal{R}_m^C = F(\mathcal{X}^C, \theta) \) change slightly with \( \mathcal{X}^C \). Observing this, we propose the canonical mean filter that generates stable features for the adaptation to realize the mean shifting.

3.2 Canonical mean filter

To strengthen the stability of \( \mathcal{R}_m^C = F(\mathcal{X}^C, \theta) \), this work proposed Canonical Mean Filter (CMF). CMF maps different support samples to a canonical form, reducing the over-sensitivity of CNAPs to the wide variation of support...
sets, and even helping remove the demand on the support sets. Note, it is important that \( \mathcal{R}_C \) represents the conditional features for adaptation on the whole dataset rather than only on a particular task. This hence means that \( \theta \) is desired to be optimized by the marginal likelihood of all tasks instead of particular tasks. The task-specific likelihood is more difficult to be optimized than the marginal likelihood of all tasks in few-shot tasks; and the former gives an inferior performance to the latter [14, 18].

Seeing this, we design CMF to realize the marginal likelihood optimization over all tasks by a kernel way to get a representative and stable \( \mathcal{R}_C \). As shown in Fig. 2, a designed attention module is used to extract the most salient features \( \Omega^{l-1} \in \mathcal{R}^{N \times C} \) at the \((l-1)_{th}\) layer \((l \in \{1, \cdots, L\})\), and then \( \Omega^{l-1} \) are averaged across all samples to fuse features

\[
\Omega^{l-1} = \frac{1}{N} \sum_{0}^{N-1} \Omega^{l-1},
\]

(5)

where \( N \) is the number of support samples in a task. \( \Omega^{l-1} \) in Eq. 5 serving as the weights, are assigned to the kernels in the \( l_{th} \) layer. Table 1 gives the structure the designed attention module.

Following CNAPs, we assume that all convolutional layers in \( F(\cdot, \theta) \) contain the same number of kernels; thus the output channel numbers are also the same. Let \( \mathcal{F}^l \) denote the input features to the \( l_{th} \) layer, \( \mathcal{F}^0 = \mathcal{X}^C \), and \( K_l \) denote the set of kernels at the \( l_{th} \) layer. CMF can be formulated as

\[
\mathcal{F}^{l+1} = (\Omega^{l-1} \otimes K^l) \otimes \mathcal{F}^l = \Omega^{l-1} \otimes (K^l \otimes \mathcal{F}^l),
\]

(6)

where \( \otimes \) denotes the Hadamard product, and \( \otimes \) denotes the convolutional operator. In Eq. 6, \( \Omega^{l-1} \) containing the average information of all samples will undermine the (biased) influence of individual samples on \( \mathcal{F}^l \). Like Dynamic Filter technology, the mean weights \( \Omega^{l-1} \) assigned to \( K^l \) adapt the conditional feature encoder to unseen data [15]. In addition, the averaging and fusion are carried out at every convolutional layer in this work rather than only at the final layer, which makes the fusion more effective in hierarchical architectures [18] and hence the ideal conditional features of \( F(\mathcal{X}^C, \theta) \) more accessible.

### 3.3 Bayesian view of CMF

This section analyzes the CMF and CNAPs from a Bayesian inference view. At the training stage, the conditional feature encoder of CNAPs is trained to maximize the the classification accuracy within a particular task. This amounts to searching for the conditional feature \( \mathcal{R}_C \) that maximizes the likelihood of \( p(\hat{Y}_k^T | \mathcal{X}_k^T, \mathcal{R}_C^m) \). Formally,

\[
\hat{\theta} = \arg \max_{\theta} p(\hat{Y}_k^T | \mathcal{X}_k^T, F(\mathcal{X}_k^C, \theta)),
\]

(7)

where \( \mathcal{R}_C^m = F(\mathcal{X}_k^C, \theta) \) and the support set \( \mathcal{X}_k^C \) is randomly chosen from \( \mathcal{X}^C \).

The obtained \( \hat{\theta} \) by Eq. (7) is task-specific, and \( F(\mathcal{X}_k^C, \hat{\theta}) \) is expected to be able to yield a task-specific feature \( \mathcal{R}_m \) that provides a good prior for parameter adaptation within the task. Unfortunately, the task-level optimization for \( \theta \) does not perform satisfactorily on few-shot tasks [14, 19], since \( F(\mathcal{X}_k^C, \hat{\theta}) \) is biased estimated and cannot provide a good conditional prior for an unseen task.

As a comparison, CMF can be formulated as a kernel method of searching the Bayesian parameter \( \theta \) that maximizes the marginal likelihood over all \( K \) tasks. Formally,

\[
\hat{\theta} = \arg \max_{\theta} \prod_{k=1}^{K} p(\hat{Y}_k^T | \mathcal{X}_k^T, F(\mathcal{X}_k^C, \theta))
\]

\[
= \arg \max_{\theta} \int \prod_{k=1}^{K} p(\hat{Y}_k^T | \mathcal{X}_k^T, F(\mathcal{X}_k^C, \theta)) p(\mathcal{X}_k^C) d\mathcal{X}_k^C
\]

\[
= \arg \max_{\theta} \frac{1}{K} \int \prod_{k=1}^{K} p(\hat{Y}_k^T | \mathcal{X}_k^T, F(\mathcal{X}_k^C, \theta)) d\mathcal{X}_k^C, \tag{8}
\]

where \( p(\mathcal{X}_k^C) \) always equals to \( \frac{1}{K} \), and \( \{\hat{Y}_k^T, \mathcal{X}_k^T\}, k = 1, \ldots, K \), denotes the target set for the \( k_{th} \) task. Once \( \hat{\theta} \) is obtained from Eq. (8), \( \mathcal{R}_C^m = F(\mathcal{X}_k^C, \hat{\theta}) \) with any arbitrary \( \mathcal{X}^C \) can provide the best average classification performance on all tasks since the influence of \( \mathcal{X}^C \) on \( \mathcal{R}_C^m \) is significantly suppressed.

Patacchiola et al. [14] proposed Deep Kernel Transfer (DKT) to approximate the kernels parameterized by \( \theta \) with the kernel of the Gaussians Process (GP) in Eq. (8), and realized the marginal optimization. Nevertheless, the GP kernels do not well approximate a wide variety of unknown kernels [18] while the unknown kernels are not in an exponential family. We designed the canonical mean filter (CMF) that yields a stable mean prior over all support sets to directly

| Table 1 | The architecture of the proposed CMF at the \((l-1)_{th}\) layer |
|---------|--------------------------------------------------|
| operators | kernel shape | output shape |
| Input | \( \mathcal{F}^{l-1} \) | \( N \times C \times W \times H \) |
| Pooling | Max Pooling | \( N \times C \) |
| FC1 | \( C \times \frac{C}{4} \) | \( N \times \frac{C}{4} \) |
| ReLU | – | – |
| FC2 | \( \frac{C}{4} \times C \) | \( N \times C \) |
| Output | \( \Omega^{l-1} \) | \( N \times C \) |

© Springer
approximate the marginal prior of the unknown kernels, $\hat{R}_m^C$, without explicitly estimating $\theta$ in Eq. (8).

4 Experiments

This section compares CNAPs-CMFe with the state-of-the-art CNAPs for multi-task classification tasks on Meta-Datasets. Three tasks are designed in Sect. 4.1 to investigate the role of feature reuse: AZS-I, AZS-II, and one-shot tasks. Like CNAPs, the models are fully trained on two different training datasets, 1) all sub-datasets of Meta-Datasets, and 2) ImageNet-1K. The trained models are then tested on all sub-datasets of Meta-Datasets as well as MNIST, CIFAR-10, and CIFAR-100 in order to evaluate the cross-domain performance. Then the clustering performance of CNAPs and CNAPs-CMF are analyzed in Sect.4.2 to investigate linear separability.

4.1 Multi-task classification

This section reports the evaluation results of AZS-II and one-shot tasks on the whole test dataset of Meta-Dataset when the models are trained on Meta-Dataset. Also, the evaluated results of CNAPs (AR) and CNAPs-CMF (AR) for one-shot tasks are reported. The result of AZS-I is reported in Table 2.

4.1.1 Training strategy

We follow the training strategy in CNAPs [4], and only 2 NVIDIA Ti-1080 GPUs with 12 G memory are used. Due to the limit of the computation resource, the auto-regressive (AR) proposed in CNAPs is not used when the model is trained on Meta-Dataset, but used when the model is trained on ImageNet-1K.

All models are trained for 110,000 episodes with the Adam [20] optimizer. The learning rate is 0.0005, and the batch size is 16. The trained model is validated per 200 episodes during the training process. For the multi-dataset classification task, a model performs better than the current model if it yields higher classification accuracy on over half of all sub-datasets.

4.1.2 Training on meta-dataset

Table 3 gives the classification performance of CNAPs-CMF on AZS-II tasks. On the Omniglot dataset, AZS-II, CNAPs-CMF outperforms CNAPs by 37.2%. Similarly, for AZS-II, CNAPs-CMF outperforms CNAPs by 38.3%, 20.7%, and 10.4% on Aircraft, Quick Draw, and MNIST datasets, respectively. When CNAPs are directly applied to AZS-II tasks, where the fixed support set is randomly chosen in the ImageNet-1k sub-dataset, the classification performance decreases significantly on Omniglot [6], Aircraft [21], Quick Draw, and MNIST. When the fixed support set is chosen in other sub-datasets, Table 11 gives a severer performance drop on all evaluated sub-datasets. CNAPs-CMF on AZS-II tasks performs only slightly inferior to CNAPs on one-shot tasks. Furthermore, we embedded the proposed CMF in the CNP-based method on the image classification task, SimpleCNAPs [22], to show its effectiveness. As shown in Table 4, SimpleCNAPs-CMF outperforms SimpleCNAPs on most sub-datasets.

For one-shot tasks, CNAPs-CMF also outperforms CNAPs by 0.3% ~ 6% on most test sub-datasets. The reason is that different support sets in the same task can yield classification performance fluctuation due to unstable conditional features.

Table 2 Multi-task image classification results. CNAPs and CNAPs-CMF are tested on AZS-I and one-shot tasks: CNAPs-One, CNAPs-AZS, and CNAPs-CMF-AZS-I

| Datasets   | CNAPs-One | CNAPs-AZS-I | CMF-AZS-I |
|-----------|-----------|-------------|-----------|
| ImageNet  | 51.3      | 50.9        | 51.2      |
| Omniglot  | 88        | 87.2        | 87.6      |
| Aircraft  | 76.8      | 75.9        | 77.2      |
| Birds     | 71.4      | 70.2        | 71.4      |
| Textures  | 62.5      | 63.1        | 63.0      |
| Quick Draw| 71.9      | 71.5        | 70.5      |
| Fungi     | 46        | 45.4        | 43.3      |
| VGG Flower| 89.2      | 87.3        | 89.6      |
| Traffic Sign | 60.1   | 58.5        | 62.0      |
| MSCOCO    | 42        | 41.7        | 43.4      |
| MNIST     | 88.6      | 87.1        | 89.1      |
| CIFAR10   | 60        | 58.7        | 65.3      |
| CIFAR100  | 48.1      | 47.6        | 50.9      |

Table 3 Multi-task image classification results. CNAPs and CNAPs-CMF are tested on AZS-II and one-shot tasks, respectively

| Datasets   | OneShot CNAPs | OneShot CNAPs-CMF | AZS-II CNAPs | AZS-II CNAPs-CMF |
|-----------|---------------|-------------------|--------------|------------------|
| ImageNet  | 51.3          | 51.6              | 50.8         | 51.2             |
| Omniglot  | 88.0          | 87.7              | 50.2         | 87.4             |
| Aircraft  | 76.8          | 77.9              | 39.1         | 77.4             |
| Birds     | 71.4          | 71.6              | 69.3         | 69.9             |
| Textures  | 62.5          | 63.9              | 64.0         | 63.7             |
| Quick Draw| 71.9          | 71.6              | 49.8         | 70.5             |
| Fungi     | 46.0          | 45.4              | 40.4         | 43.4             |
| VGG Flower| 89.2          | 89.7              | 80.5         | 89.4             |
| Traffic Sign | 60.1   | 61.3              | 59.1         | 61.4             |
| MSCOCO    | 42.0          | 44.1              | 41.9         | 43.0             |
| MNIST     | 88.6          | 89.2              | 78.7         | 89.1             |
| CIFAR10   | 60.0          | 66.1              | 58.7         | 66.3             |
| CIFAR100  | 48.1          | 50.0              | 47.6         | 51.3             |
Table 4  Multi-task image classification results. SimpleCNAPs and SimpleCNAPs-CMF (abbreviated as SCNAPs and SCNAPs-CMF) are tested on one-shot and AZS-II tasks.

| Datasets     | SCNAPs | SCNAPs-CMF | SCNAPs | SCNAPs-CMF |
|--------------|--------|------------|--------|------------|
| One-Shot     |        |            |        |            |
| ImageNet     | 58.6   | 58.9       | 57.8   | 58.3       |
| Omniglot     | 91.7   | 92.2       | 53.4   | 91.4       |
| Aircraft     | 82.4   | 81.9       | 45.6   | 81.6       |
| Birds        | 74.9   | 74.6       | 73.2   | 74.2       |
| Textures     | 67.8   | 68.4       | 66.7   | 67.5       |
| Quick Draw   | 77.7   | 78.6       | 53.1   | 77.2       |
| Fungi        | 46.9   | 47.3       | 45.8   | 46.6       |
| VGG Flower   | 90.7   | 90.8       | 86.0   | 90.1       |
| Traffic Sign | 73.5   | 73.9       | 62.3   | 73.0       |
| MSCOCO       | 46.2   | 46.8       | 45.5   | 45.8       |
| MNIST        | 93.9   | 94.0       | 83.4   | 92.9       |
| CIFAR10      | 74.3   | 75.8       | 72.2   | 74.5       |
| CIFAR100     | 60.5   | 61.2       | 59.8   | 60.3       |
| Avg.         | 72.2   | 72.7       | 61.9   | 71.9       |

The bold numbers are the highest classification accuracy in the sub-dataset.

To simply see the fluctuation, four support sets are randomly chosen from the Omniglot dataset in the same task, and Table 5 gives the classification results. For example, on Task 3, the support set $X^S_1$ gives 76.5% classification accuracy, but $X^S_4$ gives 92%, 15.5% higher than $X^S_1$. As a comparison, CMF addresses the inner-task performance fluctuation by generating stable and independent conditional features.

4.1.3 Training on single dataset

Due to the limit of computation resources, the proposed AR in CNAPs needs to be trained on the whole Meta-Dataset is not used in the following evaluation experiments. For verifying the effectiveness of the proposed CMF and maintaining the thoroughness of the whole evaluation experiments, AR is used in the following comparison experiments, in which the models are only trained on the ImageNet-1K sub-dataset.

As is given in Table 6, CNAPs-CMF (AR) outperforms CNAPs (AR) on almost all sub-datasets of Meta-Datasets.

Table 5  Classification result of CNAPs using different support sets for the same task on Omniglot. Unit: %

| Task | $X^S_1$ | $X^S_2$ | $X^S_3$ | $X^S_4$ |
|------|--------|--------|--------|--------|
| T1   | 93.5   | 88.9   | 91.5   | 90.1   |
| T2   | 81.0   | 90.0   | 85.0   | 85.5   |
| T3   | 76.5   | 80.0   | 79.5   | 92.0   |

Although the model is only trained on ImageNet-1K, the pre-trained ResNet-18 model brings the inference enough separability capability to make CNAPs (AR) can be generalized to other datasets. Also, CMF makes the conditional feature encoder $F(\cdot, \theta)$ learn better conditional features when the unseen data is input. The better representation allows the inference model $G(\phi, \mathcal{X}^T)$ to be more correctly adapted to the target distributions.

4.2 Clustering analysis of extracted features in AZS style

Similar to the reasons described in the ANIL, the adaptation part cannot be removed from the classifiers. This means that the corresponding support sets cannot be removed from the classifiers of CNAPs. Seeing this, the clustering performance of features, which are extracted by the feature extractors without adaptation, is analyzed in this section. The PCA is used to visualize the extracted features to show the better linear separability of CNAPs-CMF features.

The example figures are shown in the right column of Fig. 3 are the 512-D features extracted by CNAPs in AZS style. The 512-D features are linear inseparable, especially the features of the Texture sub-dataset. As a comparison, the features extracted by CNAPs-CMF form obvious separable clusters, which are shown in the right column and indicate the more powerful feature extractor.

For quantitative analysis, the inner-class and inter-class Mahalanobis distance between the extracted features are also given in Table 7. Firstly, PCA is used to reduce the dimensionality of the original 512-D feature embedding to 64-D for

Table 6  Classification results of CNAPs-One (AR), CNAPs-CMF-One (AR), and other SOTA methods are compared. The evaluated models are trained on ImageNet-1K only and evaluated on the whole Meta-Dataset.

| Datasets     | ProtoNet | MatchiNet | CNAP | CMF |
|--------------|----------|-----------|------|-----|
| ImageNet     | 50.5     | 45.0      | 50.6 | 50.9 |
| Omniglot     | 60.0     | 52.3      | 45.2 | 45.7 |
| Aircraft     | 53.1     | 49.0      | 36.0 | 37.2 |
| Birds        | 68.8     | 62.2      | 60.7 | 61.2 |
| Textures     | 66.6     | 64.2      | 67.5 | 67.4 |
| Quick Draw   | 49.0     | 42.9      | 42.3 | 42.9 |
| Fungi        | 39.7     | 34.0      | 30.1 | 31.3 |
| VGG Flower   | 85.3     | 80.1      | 70.7 | 71.0 |
| Traffic Sign | 47.1     | 47.8      | 53.3 | 53.1 |
| MSCOCO       | 41.0     | 35.0      | 45.2 | 45.6 |
| MNIST        | 70.4     | 72.3      | 65.2 | 66.7 |
| CIFAR10      | 53.6     | 54.3      | 65.2 | 66.7 |
| CIFAR100     |         |           | 53.6 | 54.3 |
Canonical mean filter for almost zero-shot multi-task classification

Fig. 3 The clusterable property analyses of the extracted features with CNAPs and CMFs on QuickDraw and Textures datasets. The fixed support sets are randomly selected in ImageNet-1K reducing the effect of unimportant features because Mahalanobis distance is more sensitive to the noised features than Euclidean distance. The Mahalanobis distance is used because it can reflect the distance along the data distribution [22].

As given in Table 7, for example, on Omniglot, the inner-class distance of CNAPs-CMF is 2.25, and the corresponding inner-class distance of CNAPs is 6.57. Simultaneously, the inter-class distance of CNAPs-CMF is 11.91, while the corresponding inter-class distance of CNAPs is 12.74. Obviously, the much smaller inner-class distance and similar inter-class make the extracted features, without adaptation, of CNAPs-CMF more linear separable than CNAPs. Similarly, the quantitative clustering analysis on Aircraft, Texture, and QuickDraw also verifies the effectiveness of CNAPs-CMF.

4.3 Parameter reduction and inference acceleration at the test stage

The conditional features $R_m^C$ generated by the conditional feature encoder of CNAPs-CMF is stable and can be computed with any arbitrary support set. Thus, the conditional

| Datasets   | CNAPs Inner dist | Inter dist | CNAPs-CMF Inner dist | Inter dist |
|------------|------------------|------------|----------------------|------------|
| Omniglot   | 6.57             | 12.74      | 2.25                 | 11.91      |
| Aircraft   | 4.78             | 9.23       | 1.51                 | 8.73       |
| Texture    | 5.62             | 10.54      | 1.73                 | 9.19       |
| QuickDraw  | 5.64             | 10.92      | 1.78                 | 10.18      |

| | # P (M) | E (s) | A (s) | R (s) |
|---|---------|-------|-------|-------|
| CNAPs | 21.3 | 3.5  | 20.7  | 12.3  |
| CNAPs-CMF | 12.7 | 0.0  | 0.0   | 12.3  |
features $R^C_m$ and the vectors for parameter shifting, $(y_w \otimes R^C_m)$ and $(y_b \otimes R^C_m)$, can be pre-computed in Eq. 3. Consequently, the two sub-networks, i.e., the conditional feature encoder and the adaptation network, can be removed, which hence allows for the inference to be accelerated. The two sub-networks account for about 40% parameter amount. For each sub-dataset of the Meta-Dataset, the running time of CNAPs and CNAPs-CMF for the sampled 600 tasks is computed. Table 8 gives the parameter amount and inference time. The parameter amount of CNAPs is reduced from 21.3M to 12.7M, and the inference time is reduced from $(3.5 + 20.7 + 12.3)s$ to 12.3s.

### 4.4 Ablation study

**Number of layers embedded with CMF** We changed the number of layers embedded with the proposed CMF to show the effect of CMF and the effectiveness of the layer-wise feature fusion in the hierarchical architectures. Table 9 gives the results. From Table 9, it can be observed that the more layers are embedded with CMF, the higher the performance. This shows the CMF can effectively strengthen the feature stability, alleviating the over-sensitivity of the encoder to the support set.

**Activation of CMF** The attention module designed in Table 1 is similar to Squeeze-and-Excitation (SE) [23] module. The difference between the designed attention module and the SE module is the activation function: the SE module uses Sigmoid since ReLU gives too large activated values and hence influences the optimization of the SE module; CMF uses ReLU and gets a better test classification accuracy than using Sigmoid. The comparison result is given in Table 10 since ReLU provides a better expressivity with a wider expressive range. Unlike the SE module aiming to extract sample-specific features, CMF does not suffer from the optimization problem with ReLU as it tends to extract sample-agnostic features and hence is easier to be optimized.

### 5 Conclusion

This paper investigated the role of feature reuse and adaptation in the conditional finetune-based few-shot method, CNAPs. We designed two Almost Zero-Shot (AZS) experiments, AZS-I and AZS-II. In AZS-I, a support set is randomly chosen from each sub-dataset and used on all tasks within it; in AZS-II, a support set is randomly chosen from a sub-dataset but used on all tasks of all sub-datasets. AZS-I and AZS-II showed that the support sets were unnecessary, and the performance decrease of CNAPs without adaptation originates from the over-sensitivity of the conditional features to the variation of support sets. We proposed a Canonical Mean Filter that can map varying support sets to a canonical form to deal with the over-sensitivity. The CMF can allow for the adaptation to be stopped in CNAPs without the performance drop, and the stopped adaptation leads to a 40.48% parameter reduction in the test stage. The analysis of the clustering property of extracted features by CNAPs-CMF showed that they were more separable than the features.
Canonical mean filter for almost zero-shot multi-task classification

Appendix 1 Some clustering visualization

See Fig. 4 and 5.

**Fig. 4** Clustering Visualization of $R_i^T$ (where $i \in \{1, 2, \ldots, N\}$, and $N$ is the number of test image datasets in Meta-Datasets) with the model trained on Meta-Datasets. Two-dimensional principal components are extracted by PCA.
Fig. 5 Clustering Visualization of $R_i^T$ (where $i \in \{1, 2, \ldots, N\}$, and $N$ is the number of test image datasets in Meta-Datasets) with the model trained on Meta-Datasets. Three-dimensional principal components are extracted by PCA.
Appendix 2 Different support sets selection comparison

See Table 11.

| Datasets          | Omniglot | Aircraft | Birds | Textures | Quick Draw | Fungi | VGG Flower | Traffic Sign | MSCOCO | MNIST | CIFAR10 | CIFAR100 |
|-------------------|----------|----------|-------|----------|------------|-------|------------|--------------|--------|-------|---------|----------|
| CNAPs-AZS-II      |          |          |       |          |             |       |            |              |        |       |         |          |
| ImageNet          | 31.0     | 38.7     | 45.5  | 43.0     | 42.1       | 50.1  | 46.5       | 50.1         | 50.9   | 33.2  | 51.3    | 50.8     |
| Omniglot          | 85.1     | 67.6     | 55.6  | 55.4     | 78.5       | 57.7  | 65.9       | 60.0         | 56.1   | 84.2  | 55.3    | 52.5     |
| Aircraft          | 51.6     | 74.6     | 55.5  | 54.7     | 67.2       | 59.3  | 68.9       | 55.2         | 44.7   | 53.3  | 37.2    | 36.2     |
| Birds             | 29.1     | 51.1     | 67.6  | 66.7     | 53.9       | 67.9  | 62.3       | 69.9         | 69.8   | 33.4  | 67.7    | 65.8     |
| Textures          | 49.9     | 56.7     | 58.4  | 55.6     | 56.0       | 63.8  | 60.8       | 64.6         | 65.7   | 50.6  | 64.9    | 64.4     |
| Quick Draw        | 60.4     | 60.9     | 46.3  | 44.8     | 70.6       | 54.5  | 61.4       | 57.9         | 51.7   | 63.5  | 51.9    | 50.8     |
| Fungi             | 19.0     | 30.9     | 39.6  | 35.8     | 34.0       | 42.7  | 41.1       | 42.5         | 41.2   | 21.8  | 38.2    | 37.6     |
| VGG Flower        | 63.7     | 73.6     | 74.5  | 70.5     | 82.7       | 81.8  | 83.2       | 83.4         | 80.8   | 79.6  | 79.0    | 78.7     |
| Traffic Sign      | 56.9     | 47.4     | 48.4  | 47.7     | 53.6       | 49.7  | 53.7       | 53.0         | 54.5   | 56.4  | 56.8    | 58.0     |
| MSCOCO            | 29.5     | 34.0     | 39.5  | 36.7     | 38.3       | 43.6  | 40.9       | 47.5         | 47.1   | 32.1  | 47.5    | 47.7     |
| MNIST             | 90.6     | 86.4     | 74.9  | 74.6     | 87.9       | 76.2  | 83.6       | 78.2         | 78.6   | 40.5  | 79.2    | 79.3     |
| CIFAR10           | 45.2     | 50.7     | 53.7  | 52.1     | 56.6       | 59.0  | 54.2       | 62.2         | 65.8   | 51.1  | 67.8    | 67.0     |
| CIFAR100          | 29.8     | 33.4     | 38.1  | 34.3     | 40.0       | 44.0  | 40.1       | 49.7         | 52.2   | 32.5  | 53.4    | 54.1     |
| CNAPs-CMF-AZS-II  |          |          |       |          |             |       |            |              |        |       |         |          |
| ImageNet          | 50.8     | 51.5     | 51.0  | 50.6     | 50.7       | 50.6  | 51.4       | 50.4         | 50.5   | 51.2  | 50.4    | 50.5     |
| Omniglot          | 88.0     | 86.8     | 87.8  | 88.0     | 88.2       | 88.5  | 88.0       | 87.6         | 88.0   | 87.6  | 87.9    | 87.9     |
| Aircraft          | 73.4     | 76.6     | 73.6  | 72.9     | 73.1       | 73.0  | 73.0       | 73.1         | 73.7   | 73.4  | 74.2    | 73.1     |
| Birds             | 69.2     | 69.6     | 71.3  | 70.0     | 70.4       | 69.9  | 69.9       | 70.1         | 69.1   | 70.6  | 70.2    | 70.0     |
| Textures          | 59.1     | 60.8     | 59.7  | 62.9     | 59.3       | 60.0  | 60.0       | 59.8         | 59.6   | 60.2  | 59.9    | 59.5     |
| Quick Draw        | 70.4     | 70.0     | 69.7  | 70.1     | 70.6       | 70.4  | 69.5       | 70.0         | 69.7   | 70.3  | 70.7    | 69.9     |
| Fungi             | 43.4     | 43.4     | 43.4  | 42.1     | 43.6       | 44.2  | 43.6       | 43.8         | 44.3   | 43.5  | 43.0    | 44.1     |
| VGG Flower        | 87.5     | 87.9     | 87.9  | 88.5     | 87.5       | 87.9  | 88.2       | 88.0         | 87.3   | 88.4  | 88.1    | 88.1     |
| Traffic Sign      | 60.0     | 61.8     | 61.1  | 60.1     | 60.2       | 61.5  | 60.7       | 62.1         | 61.2   | 61.6  | 61.7    | 61.4     |
| MSCOCO            | 45.4     | 43.9     | 45.5  | 44.0     | 45.3       | 44.7  | 44.4       | 45.6         | 44.4   | 44.2  | 44.1    | 44.1     |
| MNIST             | 89.0     | 88.1     | 88.9  | 88.5     | 88.5       | 88.9  | 88.5       | 88.9         | 89.5   | 89.2  | 88.8    |          |
| CIFAR10           | 62.5     | 62.7     | 62.7  | 62.6     | 62.5       | 62.9  | 63.2       | 62.2         | 62.8   | 63.6  | 63.8    | 63.6     |
| CIFAR100          | 49.7     | 48.8     | 49.4  | 47.1     | 47.8       | 47.5  | 48.3       | 47.9         | 47.7   | 48.1  | 47.8    | 49.3     |

The bold numbers are the highest classification accuray in the sub-dataset.
Acknowledgements The authors declare that they have no conflict of interest.

Data availability The data will be available on reasonable requests.

References

1. Vinyals O, Blundell C, Lillicrap T, Kavukcuoglu K, Wierstra D (2016) Matching networks for one shot learning. In: Advances in Neural Information Processing Systems, vol 29. https://proceedings.neurips.cc/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf
2. Finn C, Abbeel P, Levine S (2017) Model-agnostic meta-learning for fast adaptation of deep networks. http://arxiv.org/abs/1703.03400
3. Raghu A, Raghu M, Bengio S, Vinyals O (2020) Rapid learning or feature reuse? Towards understanding the effectiveness of maml. In: International Conference on Learning Representations. https://openreview.net/forum?id=rkgMkCEtPB
4. Requeima J, Gordon J, Bronskill J, Nowozin S, Turner RE (2019) Fast and flexible multi-task classification using conditional neural adaptive processes. In: Advances in Neural Information Processing Systems 32, pp 7957–7968
5. Triantafillou E, Zhu T, Dumoulin V, Lamblin P, Evci U, Xu K, Goroshin R, Gelada C, Swersky K, Manzagol P-A, Larochelle H (2020) Meta-dataset: a dataset of datasets for learning to learn from few examples. In: International Conference on Learning Representations
6. Lake BM, Salakhutdinov R, Tenenbaum JB (2015) Human-level concept learning through probabilistic program induction. Science 350(6266):1332–1338. https://doi.org/10.1126/science.aab3050. https://science.sciencemag.org/content/350/6266/1332.full.pdf
7. Garnelo M, Rosenbaum D, Maddison CJ, Ramalho T, Saxton D, Shanahan M, Teh YW, Rezende DJ, EslamiSM (2018) Conditional neural processes. In: International Conference on Machine Learning
8. Garnelo M, Schwarz J, Rosenbaum D, Viola F, Rezende DJ, Eslami S, Teh YW (2018) Neural processes. Preprint at http://arxiv.org/abs/1807.01622
9. Kim H, Mnih A, Schwarz J, Garnelo M, Eslami A, Rosenbaum D, Vinyals O, Teh YW (2019) Attentive neural processes. Preprint at http://arxiv.org/abs/1901.05761
10. Gordon J, Bruinsma WP, Foong AY, Requeima J, Dubois Y, Turner RE (2019) Convolutional conditional neural processes. Preprint at http://arxiv.org/abs/1910.13556
11. Hayashi T, Fujita H (2022) OCFSNP: self-supervised one-class classification approach using feature-slice prediction subtask for feature data. Soft Comput 26(19):10127–10149
12. Hayashi T, Cimr D, Studnička F, Fujita H, Bušovský D, Cimler R (2022) OCSTN: one-class time-series classification approach using a signal transformation network into a goal signal. Inf Sci 614:71–86
13. Hayashi T, Fujita H (2021) Cluster-based zero-shot learning for multivariate data. J Ambient Intell Humaniz Comput 12(2):1897–1911
14. Patacchiola M, Turner J, Crowley EJ, Storkey A (2020) Bayesian meta-learning for the few-shot setting via deep kernels. In: Advances in Neural Information Processing Systems
15. Chen Y, Dai X, Liu M, Chen D, Liu Z (2020) Dynamic convolution: attention over convolution kernels. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
16. Perez E, Strub F, de Vries H, Dumoulin V, Courville AC (2018) Film: visual reasoning with a general conditioning layer. In: AAAI
17. He K, Zhang X, Ren S (2015) Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. pp 770–778
18. Robbins H (2020) An empirical Bayes approach to statistics, pp 157–164. https://doi.org/10.1525/9780520313880-015
19. Liu Y, Schiele B, Sun Q (2020) Ensemble of epoch-wise empirical Bayes for few-shot learning. In: European Conference on Computer Vision (ECCV)
20. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio Y, LeCun Y (eds) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings. Preprint at http://arxiv.org/abs/1412.6980
21. Maji S, Kanan J, Rahtu E, Blaschko M, Vedaldi A (2013) Fine-grained visual classification of aircraft. Technical report
22. Bateni P, Goyal R, Masrani V, Wood F, Sigal L (2020) Improved few-shot visual classification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
23. Hu J, Shen L, Sun G (2017) Squeeze-and-excitation networks. In: IEEE Conference on Computer Vision and Pattern Recognition

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.