Selecting Relevant Features from a Multi-domain Representation for Few-shot Classification

Nikita Dvornik, Cordelia Schmid, Julien Mairal

Selecting Relevant Features from a Multi-domain Representation for Few-shot Classification

Nikita Dvornik[1], Cordelia Schmid[2], and Julien Mairal[1]

[1] Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
[2] Inria, École normale supérieure, CNRS, PSL Research Univ., 75005 Paris, France
`firstname.lastname@inria.fr`

Abstract. Popular approaches for few-shot classification consist of first learning a generic data representation based on a large annotated dataset, before adapting the representation to new classes given only a few labeled samples. In this work, we propose a new strategy based on feature selection, which is both simpler and more effective than previous feature adaptation approaches. First, we obtain a multi-domain representation by training a set of semantically different feature extractors. Then, given a few-shot learning task, we use our multi-domain feature bank to automatically select the most relevant representations. We show that a simple non-parametric classifier built on top of such features produces high accuracy and generalizes to domains never seen during training, leading to state-of-the-art results on MetaDataset and improved accuracy on *mini*-ImageNet.

Keywords: Image Recognition, Few-shot Learning, Feature Selection

1 Introduction

Convolutional neural networks [21] (CNNs) have become a classical tool for modeling visual data and are commonly used in many computer vision tasks such as image classification [19], object detection [8, 24, 36], or semantic segmentation [8, 25, 38]. One key of the success of these approaches relies on massively labeled datasets such as ImageNet [39] or COCO [23]. Unfortunately, annotating data at this scale is expensive and not always feasible, depending on the task at hand. Improving the generalization capabilities of deep neural networks and removing the need for huge sets of annotations is thus of utmost importance.

This ambitious challenge may be addressed from different perspectives, such as large-scale unsupervised learning [3], self-supervised learning [6, 13], or by developing regularization techniques dedicated to deep networks [1, 51]. An alternative solution is to use data that has been previously annotated for a different task than the one considered, for which only a few annotated samples may be available. This approach is particularly useful if the additional data is related to the new task [52, 54], which is unfortunately not known beforehand. How to use effectively this additional data is then an important subject of ongoing research [9, 46, 54]. In this paper, we propose to use a multi-domain image representation, *i.e.*, an exhaustive set of semantically different features.
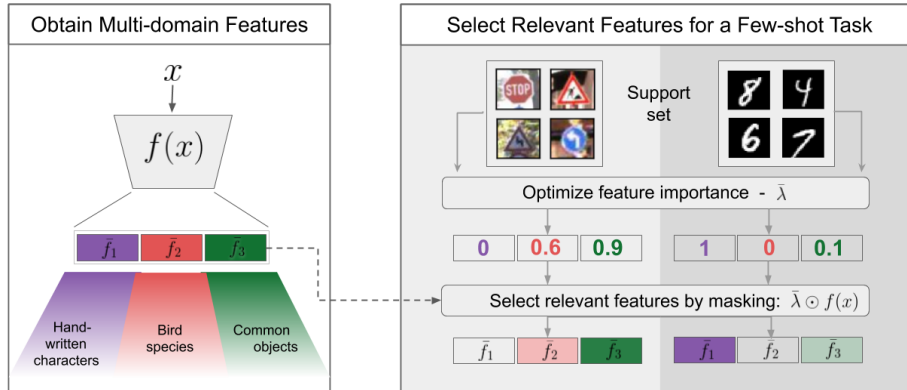
Fig. 1: **Illustration of our approach.** (Left) First, we obtain a multi-domain feature representation, consisting of feature blocks with different semantics. (Right) Given a few-shot task, we select only the relevant feature blocks from the multi-domain representation, by optimizing masking parameters λ on the support set.

Then, by automatically selecting only relevant feature subsets from the multi-domain representation, we show how to successfully solve a large variety of target tasks.

Specifically, we are interested in few-shot classification, where a visual model is first trained from scratch, *i.e.* starting from randomly initialized weights, using a large annotated corpus. Then, we evaluate its ability to transfer the knowledge to new classes, for which only very few annotated samples are provided. Simply fine-tuning a convolutional neural network on a new classification task has been shown to perform poorly [9]. This has motivated the community to develop dedicated techniques, allowing effective adaptation with few samples.

Few-shot classification methods typically operate in two stages, consisting of first pre-training a general feature extractor and then building an adaptation mechanism. A common way to proceed is based on meta-learning [9, 33, 42, 44, 45, 47], which is a principle to learn how to adapt to new learning problems. That is, a parametric adaptation module (typically, a neural network) is trained to produce a classifier for new categories, given only a few annotated samples [12, 32, 40]. To achieve this goal, the large training corpus is split into smaller few-shot classification problems [9, 33, 47], that are used to train the adaptation mechanism. Training in such episodic fashion is advocated to alleviate overfitting and improve generalization [9, 33, 47]. However, a more recent work [4] demonstrates that using large training set to train the adapter is not necessary, *i.e.* a linear classifier with similar accuracy can be trained directly from new samples, on top of a fixed feature extractor. Finally, it has been shown that adaptation is not necessary at all [7]; using a non-parametric prototypical classifier [28] combined with a properly regularized feature extractor can achieve better accuracy than recent meta-learning baselines [7, 41]. These results suggest that on standard few-shot

learning benchmarks [33, 35], the little amount of samples in a few-shot task is not enough to learn a meaningful adaptation strategy.

To address the shortcomings of existing few-shot benchmarks, the authors of [46] have proposed MetaDataset, which evaluates the ability to learn across different visual domains and to generalize to new data distributions at test time, given few annotated samples. While methods based solely on pre-trained feature extractors [41] can achieve good results only on test datasets that are similar to the training ones, the adaptation technique [37] performs well across test domains. The method not only predicts a new few-shot classifier but also adapts the filters of a feature extractor, given an input task. The results thus suggest that feature adaptation may be in fact useful to achieve better generalization.

In contrast to these earlier approaches, we show that feature adaptation can be replaced by a simple feature selection mechanism, leading to better results in the cross-domain setup of [46]. More precisely, we propose to leverage a multi-domain representation – a large set of semantically different features that captures different modalities of a training set. Rather than adapting existing features to a new few-shot task, we propose to select features from the multi-domain representation. We call our approach SUR which stands for Selecting from Universal Representations. To be more clear, we say universal because SUR could be applied not only to multi-domain representations, but to any set of representations that are semantically different. In contrast to standard adaptation modules [30, 37, 46] learned on the training set, selection is performed directly on new few-shot tasks using gradient descent. Approaching few-shot learning with SUR has several advantages over classical adaptation techniques. First, it is simple by nature, *i.e.* selecting features from a fixed set is an easier problem than learning a feature transformation, especially when few annotated images are available. Second, learning an adaptation module on the meta-training set is likely to generalize only to similar domains. In contrast, the selection step in our approach is decoupled from meta-training, thus, it works equally well for any new domain. Finally, we show that our approach achieves better results than current state-of-the-art methods on popular few-shot learning benchmarks. In summary, this work makes the following contributions:

– We propose to tackle few-shot classification by selecting relevant features from a multi-domain representation. While multi-domain representations can be built by training several feature extractors or using a single neural network, the selection procedure is implemented with gradient descent.
– We show that our method outperforms existing approaches in in-domain and cross-domain few-shot learning and sets new state-of-the-art result on MetaDataset [46] and improves accuracy on *mini*-ImageNet [33].

Our implementation is available at https://github.com/dvornikita/SUR.

2   Related Work

In this section, we now present previous work on few-shot classification and multi-domain representations, which is a term first introduced in [2].

Few-shot classification. Typical few-shot classification problems consist of two parts called meta-training and meta-testing [4]. During the meta-training stage, one is given a large-enough annotated dataset, which is used to train a predictive model. During meta-testing, novel categories are provided along with few annotated examples. The goal is to evaluate the ability of the predictive model to adapt and perform well on these new classes.

Typical few-shot learning algorithms [11, 12, 32] first pre-train the feature extractor by supervised learning on the meta-training set. Then, they use meta-learning [42, 45] to train an adaptation mechanism. For example, in [12, 32], adaptation consists of predicting the weights of a classifier for new categories, given a small few-shot training set. The work of [30] goes beyond the adaptation of a single layer on top of a fixed feature extractor, and additionally generates FiLM [31] layers that modify convolutional layers. Alternatively, the work of [4] proposes to train a linear classifier on top of the features directly from few samples from new categories. In the same line of work, [22] performs implanting, *i.e.* learning new convolutional filters within the existing CNN layers.

Other methods do not perform adaptation at all. It has been shown in [7, 22, 41] that training a regularized CNN for classification on the meta-training set and using these features directly with a nearest centroid classifier produces state-of-the-art few-shot accuracy. To obtain a robust feature extractor, [7] distills an ensemble of networks into a single extractor to obtain low-variance features.

Finally, the methods [37, 41] are the most relevant to our work as they also tackle the problem of cross-domain few-shot classification [46]. In [37], the authors propose to adapt each hidden layer of a feature extractor for a new task. They first obtain a task embedding and use conditional neural process [10] to generate parameters of modulation FiLM [31] layers, as well as weights of a classifier for new categories. An adaptation-free method [41] instead trains a CNN on ImageNet, while optimizing for high validation accuracy on other datasets, using hyper-parameter search. When tested on domains similar to ImageNet, the method demonstrates the highest accuracy, however, it is outperformed by adaptation-based methods when tested on other data distributions [41].

Multi-domain Representations. A multi-domain representation (introduced as "universal representation" in [2]) refers to an image representation that works equally well for a large number of visual domains. The simplest way to obtain a multi-domain representation is to train a separate feature extractor for each visual domain and use only the appropriate one at test time. To reduce the computational footprint, [2] investigates if a single CNN can be useful to perform image classification on very different domains. To achieve this goal, the authors propose to share most of the parameters between domains during training and have a small set of parameters that are domain-specific. Such adaptive feature sharing is implemented using conditional batch normalization [16], *i.e.* there is a separate set of batch-norm parameters for every domain. The work of [34] extends the idea of domain-specific computations in a single network and proposes universal parametric network families, which consist of two parts: 1) a CNN

feature extractor with universal parameters shared across all domains, and 2) domain-specific modules trained on top of universal weights to maximize the performance on that domain. It has been found important [34] to adapt both shallow and deep layers in a neural network in order to successfully solve multiple visual domains. We use the method of this paper in our work when training a parametric network family to produce a multi-domain representation. In contrast, instead of parallel adapters, in this work, we use much simpler FiLM layers for domain-specific computations. Importantly, parametric networks families [34] and FiLM [31] adapters only provide a way to efficiently compute multi-domain representation; they are not directly useful for few-shot learning. However, using our SUR strategy on this representation produces a useful set of features leading to state-of-the-art results in few-shot learning.

3 Proposed Approach

We now present our approach for few-shot learning, starting with preliminaries.

3.1 Few-Shot Classification with Nearest Centroid Classifier

The goal of few-shot classification is to produce a model which, given a new learning task and a few labeled examples, is able to generalize to unseen examples for that task. In other words, the model learns from a small training set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n_S}$, called a *support set*, and is evaluated on a held-out test set $\mathcal{Q} = \{(x_j^*, y_j^*)\}_{j=1}^{n_Q}$, called a *query set*. The (x_i, y_i)'s represent image-label pairs while the pair $(\mathcal{S}, \mathcal{Q})$ represents the few-shot task. To fulfill this objective, the problem is addressed in two steps. During the meta-training stage, a learning algorithm receives a large dataset $\mathcal{D}_b$, where it must learn a general feature extractor $f(\cdot)$. During the meta-testing stage, one is given a target dataset $\mathcal{D}_t$, used to repeatedly sample few-shot tasks $(\mathcal{S}, \mathcal{Q})$. Importantly, meta-training $(\mathcal{D}_b)$ and meta-testing $(\mathcal{D}_t)$ datasets have no categories in common.

During the meta-testing stage, we use feature representation $f(\cdot)$ to build a nearest centroid classifier (NCC), similar to [28, 44]. Given a support set $\mathcal{S}$, for each category present in this set, we build a class centroid by averaging image representations belonging to this category:

$$c_j = \frac{1}{|\mathcal{S}_j|} \sum_{i \in \mathcal{S}_j} f(x_i), \quad \mathcal{S}_j = \{k : y_k = j\}, \quad j = 1, ..., C. \tag{1}$$

To classify a sample x, we choose a distance function $d(f(x), c_j)$ to be negative cosine similarity, as in [4, 13], and assign the sample to the closest centroid c_j.

3.2 Method

With the NCC classifier defined above, we may now formally define the concept of multi-domain representation and the procedure for selecting relevant features.

Typically, the meta-training set is used to train a set of K feature extractors $\{f_i(\cdot)\}_{i=1}^K$ that form a multi-domain set of features. Each feature extractor maps an input image x into a d-dimensional representation $f_i(x) \in \mathbb{R}^d$. These features should capture different types of semantics and can be obtained in various manners, as detailed in Section 3.3.

Parametric Multi-domain Representations. One way to transform a multi-domain set of features into a vectorized multi-domain representation $f(x)$ is, by concatenating all image representations from this set (with or without $l2$-normalization). As we show in the experimental section, directly using such $f(x)$ for classification with NCC does not work well as many irrelevant features for a new task are present in the representation. Therefore, we are interested in implementing a selection mechanism. In order to do so, we define a selection operation $\odot$ as follows, given a vector λ in $\mathbb{R}^K$:

$$\lambda \odot f(x) = \lambda \cdot \begin{bmatrix} \hat{f}_1(x) \\ \vdots \\ \hat{f}_K(x) \end{bmatrix} = \begin{bmatrix} \lambda_1 \cdot \hat{f}_1(x) \\ \vdots \\ \lambda_K \cdot \hat{f}_K(x) \end{bmatrix} = f_\lambda(x), \tag{2}$$

where $\hat{f}_i(x)$ is simply $f_i(x)$ after ℓ_2-normalization. We call $f_\lambda(x) \in \mathbb{R}^{K \cdot d}$ a parametrized multi-domain representation, as it contains information from the whole multi-domain set but the exact representation depends on the selection parameters λ. Using this mechanism, it is possible to select various combinations of features from the multi-domain representation by setting more than one λ_i to non-zero values.

Finding optimal selection parameters. Feature selection is performed during meta-testing by optimizing a probabilistic model, leading to optimal parameters λ, given a support set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n_S}$ of a new task.

Specifically, we consider the NCC classifier from Section 3.1, using $f_\lambda(x)$ instead of $f(x)$, and introduce the likelihood function

$$p(y = l|x) = \frac{\exp(-d(f_\lambda(x), c_l))}{\sum_{j=1}^{n_S} \exp(-d(f_\lambda(x), c_j))}. \tag{3}$$

Our goal is then to find optimal parameters λ that maximize the likelihood on the support set, which is equivalent to minimizing the negative log-likelihood:

$$L(\lambda) = \frac{1}{n_S} \sum_{i=1}^{n_S} [-\log(p(y = y_i|x_i))] \tag{4}$$

$$= \frac{1}{n_S} \sum_{i=1}^{n_S} \left[\log \sum_{j=1}^{C} \exp(\cos(f_\lambda(x_i), c_j)) - \cos(f_\lambda(x_i), c_{y_i}) \right]. \tag{5}$$

This objective is similar to the one of [49] and encourages large lambda values to be assigned to representations where intra-class similarity is high while the inter-class similarity is low. In practice, we optimize the objective by performing several steps of gradient descent. The proposed procedure is what we call SUR.

It is worth noting that nearest centroid classifier is a simple non-parametric model with limited capacity, which only stores a single vector to describe a class. Such limited capacity becomes an advantage when only a few annotated samples are available, as it effectively prevents overfitting. When training and testing across similar domains, SUR is able to select from the multi-domain representation the features optimized for each visual domain. When the target domain distribution does not match any of the train distribution, it is nevertheless able to adapt a few parameters λ to the target distribution. In such a sense, our method performs a limited form of adaptation, with few parameters only, which is reasonable given that the target task has only a small number of annotated samples.

Sparsity in selection. As said above, our selection algorithm is a form of weak adaptation, where parameters λ are adjusted to optimize image representation, given an input task. Selection parameters λ_i are constrained to be in $[0, 1]$. However, as we show in the experimental section, the resulting λ vector is sparse in practice, with many entries equal to 0 or 1. This empirical behavior suggests that our algorithm indeed performs selection of relevant features – a simple and robust form of adaptation. To promote further sparsity, one may use an additional sparsity-inducing penalty such as ℓ_1 during the optimization [26]; however, our experiments show that doing so is not necessary to achieve good results.

3.3   Obtaining Multi-Domain Representations

In this section, we explain how to obtain a multi-Domain set of feature extractors $\{f_i(\cdot)\}_i^K$ when one or multiple domains are available for training. Three variants are used in this paper, which are illustrated in Figure 2.

Multiple training domains. In the case of multiple training domains, we assume that K different datasets (including ImageNet [39]) are available for building a multi-domain representation. We start with a straightforward solution and train a feature extractor $f_i(\cdot)$ for each visual domain independently. That results in a desired multi-domain set $\{f_i(\cdot)\}_i^K$. A multi-domain representation is then computed by concatenating the output set, as illustrated in Figure 2(a).

To compute multi-domain representations with a single network, we use parametric network families proposed in [34]. We follow the original paper and first train a general feature extractor – ResNet – using our training split of ImageNet, and then freeze the network's weights. For each of the $K-1$ remaining datasets, task-specific parameters are learned by minimizing the corresponding training loss. We use FiLM [31] layers as domain-specific modules and insert them after each batch-norm layer in the ResNet. We choose to use FiLM layers
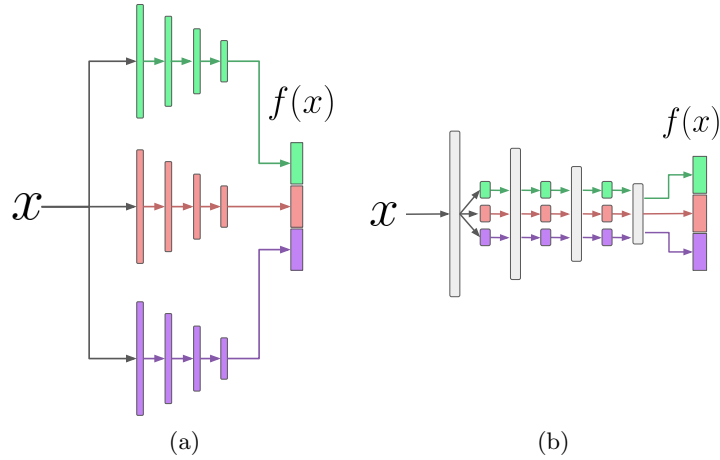
Fig. 2: Different ways of obtaining a multi-domain representation. (a) A single image is embedded with multiple domain-specific networks. (b) Using a parametric network family [34] to obtain multi-domain representations. Here, the gray blocks correspond to shared computations and colored blocks correspond to domain-specific FiLM [31] layers. Gray blocks and arrows indicate shared layers and computation flow respectively, while domain specific ones are shown in corresponding colors. Best viewed in color.

over originally proposed parallel adapters [34] because FiLM is much simpler, i.e. performs channel-wise affine transformation, and contain much fewer parameters. This helps to avoid overfitting on small datasets. In summary, each of the K datasets has its own set of FiLM layers and the base ResNet with a set of domain-specific FiLM layers constitutes the multi-domain set of extractors $\{f_i(\cdot)\}_i^K$. To compute the features of i-th domain $f_i(x)$, we forward the input image x trough the ResNet where all the intermediate activations are modulated with the FiLM layers trained on this domain, as illustrated in Figure 2(b). Using parametric network families instead of K separate networks to obtain a multi-domain representation reduces the number of stored parameters roughly in K times. However, to be actually useful for few-shot learning, such representation must be processed by our SUR approach, as described in Section 3.2

4 Experiments

We now present the experiments to analyze the performance of our selection strategy, starting with implementation details.

4.1 Datasets and Experiments Details

Datasets. We use *mini*-ImageNet [33] and Meta-Dataset [46] to evaluate the proposed approach. The *mini*-ImageNet [33] dataset consists of 100 categories

(64 for training, 16 for validation, 20 for testing) from the original ImageNet [39] dataset, with 600 images per class. Since all the categories come from the same dataset, we use *mini*-ImageNet to evaluate our feature selection strategy in single-domain few-shot learning. During testing on *mini*-ImageNet, we measure performance over tasks where only 1 or 5 images (shots) per category are given for adaptation and the number of classes in a task is fixed to 5, *i.e.* 5-way classification. All images are resized to 84×84, as suggested originally by [33].

Meta-Dataset [46] is much larger than previous few-shot learning benchmarks and it is actually a collection of multiple datasets with different data distributions. It includes `ImageNet` [39], `Omniglot` [20], `Aircraft` [27], `CU-Birds` [48], Describable `Textures` [5], `Quick Draw` [17], `Fungi` [43], `VGG-Flower` [29], `Traffic Sign` [15] and `MSCOCO` [23]. A short description of each dataset is contained in Appendix. `Traffic Sign` and `MSCOCO` datasets are reserved for testing only, while all other datasets have their corresponding train, val and test splits. To better study out-of-training-domain behavior, we follow [37] and add 3 more testing datasets, namely `MNIST` [50] `CIFAR10` [18], and `CIFAR100` [18]. Here, the number of shots and ways is not fixed and varies from one few-shot task to another. As originally suggested [46], all images are resized to 84×84 resolution.

Implementation Details. When experimenting with Meta-Dataset, we follow [37] and use ResNet18 [14] as feature extractor. The training detail for each dataset are described in Appendix. To report test results on Meta-Dataset, we perform an independent evaluation for each of the 10 provided datasets, plus for 3 extra datasets as suggested by [37]. We follow [46] and sample 600 tasks for evaluation on each dataset within Meta-Dataset.

When experimenting with *mini*-ImageNet, we follow popular works [11, 22, 30] and use ResNet12 [30] as a feature extractor. Corresponding training details are reported in Appendix. During testing, we use *mini*-ImageNet's test set to sample 1000 5-way classification tasks. We evaluate scenarios where only 1 or 5 examples (shots) of each category is provided for training and 15 for evaluation. On both datasets, during meta-training, we use cosine classifier with learnable softmax temperature [4]. During testing, classes and corresponding train/test examples are sampled at random. For all our experiments, we report the mean accuracy (in %) over all test tasks with 95% confidence interval.

Feature Selection. To perform feature selection from the multi-domain representation we optimize the selection parameter λ (defined in Eq. 2) to minimize NCC classification loss (Eq. 4) on the support set. Each individual scalar weight λ_i is kept between 0 and 1 using sigmoid function, *i.e.* $\lambda_i = \mathrm{sigmoid}(\alpha_i)$. All α_i are initialized with zeros. We optimize the parameters α_i using gradient descent for 40 iterations. At each iteration, we use the whole support set to build nearest centroid classifier, and then we use the same set of examples to compute the loss, given by Eq. 4. Then, we compute gradients w.r.t $[\alpha_1, ..., \alpha_K]$ and use Adadelta [53] optimizer with learning rate 10^2 to perform parameter updates.

Dataset	Features trained on:							
	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower
ImageNet	**56.3±1.0**	18.5±0.7	21.5±0.8	23.9±0.8	26.1±0.8	23.1±0.8	31.2±0.8	24.3±0.8
Omniglot	67.5±1.2	**92.4±0.5**	55.2±1.3	59.5±1.3	48.4±1.3	80.0±0.9	59.7±1.2	54.2±1.4
Aircraft	50.4±0.9	17.0±0.5	**85.4±0.5**	30.9±0.7	23.9±0.6	25.2±0.6	33.7±0.8	25.1±0.6
Birds	**71.7±0.8**	13.7±0.6	18.0±0.7	64.7±0.9	20.2±0.7	17.9±0.6	40.7±0.9	24.5±0.8
Textures	**70.2±0.7**	30.6±0.6	33.1±0.6	37.1±0.6	57.3±0.7	38.5±0.7	50.4±0.7	45.4±0.8
Quick Draw	52.4±1.0	50.3±1.0	36.0±1.0	38.8±1.0	35.7±0.9	**80.7±0.6**	35.4±1.0	39.4±1.0
Fungi	39.1±1.0	10.5±0.5	14.0±0.6	21.2±0.7	15.5±0.7	13.0±0.6	**62.7±0.9**	22.6±0.8
VGG Flower	**84.3±0.7**	24.8±0.7	44.6±0.8	57.2±0.8	42.3±0.8	36.9±0.8	76.1±0.8	77.1±0.7
Traffic Sign	**63.1±0.8**	44.0±0.9	57.7±0.8	61.7±0.8	55.2±0.8	50.2±0.8	53.5±0.8	57.9±0.8
MSCOCO	**52.8±1.0**	15.1±0.7	21.2±0.8	22.5±0.8	25.8±0.9	19.9±0.7	29.3±0.9	27.3±0.9
MNIST	77.2±0.7	**90.9±0.5**	69.5±0.7	74.2±0.7	55.9±0.8	86.2±0.6	69.4±0.7	66.9±0.7
CIFAR 10	**66.3±0.8**	33.0±0.7	37.8±0.7	39.3±0.7	39.2±0.7	36.1±0.7	33.6±0.7	38.2±0.7
CIFAR 100	**55.7±1.0**	14.9±0.7	22.5±0.8	25.6±0.8	24.1±0.8	21.4±0.7	22.2±0.8	26.5±0.9

Table 1: **Performance of feature extractors trained with different datasets on Meta-Dataset.** The first column indicates the dataset in Meta-Dataset, the first row gives the dataset, used to pre-train the feature extractor. The body of the table shows feature extractors' accuracy on few-shot classification, when applied with NCC. The average accuracy and 95% confidence intervals computed over 600 few-shot tasks. The numbers in bold indicate that a method has the best accuracy per dataset.

4.2   Cross-Domain Few-shot Classification

In this section, we evaluate the ability of SUR to handle different visual domains in MetaDataset [46]. First, we motivate the use of multi-domain representations and show the importance of feature selection. Then, we evaluate the proposed strategy against important baselines and state-of-the-art few-shot algorithms.

Evaluating domain-specific feature extractors. MetaDataset includes 8 datasets for training, *i.e.* ImageNet, Omniglot, Aircraft, CU-Birds, Textures, Quick Draw, Fungi, VGG-Flower. We treat each dataset as a separate visual domain and obtain a multi-domain set of features by training 8 domain-specific feature extractors, *i.e.* a separate ResNet18 for each dataset. Each extractor is trained independently, with its own training schedule specified in Appendix. We test the performance of each feature extractor (with NCC) on every test dataset specified in Section 4.1, and report the results in Table 1. Among the 8 datasets seen during training, 5 datasets are better solved with their own features, while 3 other datasets benefit more from ImageNet features. In general, ImageNet features suits 8 out of 13 test datasets best, while 5 others require a different feature extractor for better accuracy. Such results suggest that none of the domain-specific feature extractors alone can perform equally well on all the datasets simultaneously. However, using the whole multi-domain feature set to select appropriate representations would lead to superior accuracy.

Evaluating feature selection. We now employ SUR – our strategy for feature selection – as described in Sec. 3.2. A parametrized multi-domain representation
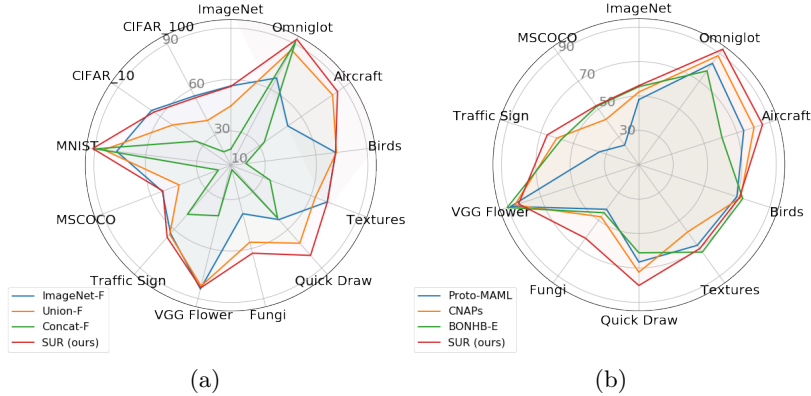
Fig. 3: **Performance of different few-shot methods on MetaDataset**. (a) Comparison of our selection strategy to baselines. The chart is generated from the table in Appendix. (b) Comparing our selection strategy to state-of-the-art few-shot learning methods. The chart is generated from Table 2. The axes indicate the accuracy of methods on a particular dataset. Each color corresponds to a different method.

is obtained from a multi-domain feature set by concatenation. It is then multiplied by the selection parameters λ, that are being optimized for each new few-shot task, following Section 4.1. We ran this procedure on all 13 testing datasets and report the results in Figure 3 (a). We compare our method with the following baselines: a) using a single feature extractor pre-trained on `ImageNet` split of MetaDataset (denoted "ImageNet-F"), b) using a single feature extractor pre-trained on the union of 8 training splits in MetaDataset (denoted "Union-F") and c) manually setting all $\lambda_i = 1$, which corresponds to simple concatenation and (denoted "Concat-F"). It is clear that the features provided by SUR have much better performance than any of the baselines on seen and unseen domains.

Comparison to other approaches. We now compare SUR against state-of-the-art few-shot methods and report the results in Table 2. The results on `MNIST`, `CIFAR 10` and `CIFAR 100` datasets are missing for most of the approaches because those numbers were not reported in the corresponding original papers. Comparison to the best-performing methods on common datasets is summarized in Figure 3 (b). We see that SUR demonstrated state-of-the-art results on 9 out of 13 datasets. BOHNB-E [41] outperforms our approach on `Birds`, `Textures` and `VGG Flowers` datasets. This is not surprising since these are the only datasets that benefit from ImageNet features more than from their own (see Table 1); and BOHNB-E [41] is essentially an ensemble of multiple ImageNet-pretrained networks. When tested outside the training domain, SUR consistently outperforms CNAPs [37] – the state-of-the-art adaptation-based method. Moreover, SUR shows the best results on all 5 datasets never seen during training.

Test Dataset	ProtoNet [44]	MAML [9]	Proto-MAML [46]	CNAPs [37]	BOHB-E [41]	SUR (ours)	SUR-pf (ours)	SUR-merge (ours)
ImageNet	44.5±1.1	32.4±1.0	47.9±1.1	52.3±1.0	55.4±1.1	**56.3±1.1**	**56.4±1.2**	**57.2±1.1**
Omniglot	79.6±1.1	71.9±1.2	82.9±0.9	88.4±0.7	77.5±1.1	**93.1±0.5**	88.5±0.8	**93.2±0.8**
Aircraft	71.1±0.9	52.8±0.9	74.2±0.8	80.5±0.6	60.9±0.9	85.4±0.7	79.5±0.8	**90.1±0.8**
Birds	67.0±1.0	47.2±1.1	70.0±1.0	72.2±0.9	73.6±0.8	71.4±1.0	76.4±0.9	**82.3±0.8**
Textures	65.2±0.8	56.7±0.7	67.9±0.8	58.3±0.7	**72.8±0.7**	71.5±0.8	**73.1±0.7**	**73.5±0.7**
Quick Draw	65.9±0.9	50.5±1.2	66.6±0.9	72.5±0.8	61.2±0.9	**81.3±0.6**	75.7±0.7	**81.9±1.0**
Fungi	40.3±1.1	21.0±1.0	42.0±1.1	47.4±1.0	44.5±1.1	63.1±1.0	48.2±0.9	**67.9±0.9**
VGG Flower	86.9±0.7	70.9±1.0	88.5±1.0	86.0±0.5	**90.6±0.6**	82.8±0.7	**90.6±0.5**	88.4±0.9
Traffic Sign	46.5±1.0	34.2±1.3	34.2±1.3	60.2±0.9	57.5±1.0	**70.4±0.8**	65.1±0.8	67.4±0.8
MSCOCO	39.9±1.1	24.1±1.1	24.1±1.1	42.6±1.1	**51.9±1.0**	52.4±1.1	**52.1±1.0**	51.3±1.0
MNIST	-	-	-	92.7±0.4	-	**94.3±0.4**	93.2±0.4	90.8±0.5
CIFAR 10	-	-	-	61.5±0.7	-	**66.8±0.9**	**66.4±0.8**	**66.6±0.8**
CIFAR 100	-	-	-	50.1±1.0	-	56.6±1.0	57.1±1.0	**58.3±1.0**

Table 2: **Comparison to existing methods on Meta-Dataset.** The first column indicates the of a dataset used for testing. The first row gives a name of a few-shot algorithm. The body of the table contains average accuracy and 95% confidence intervals computed over 600 few-shot tasks. The numbers in bold have intersecting confidence intervals with the most accurate method.

Multi-domain representations with parametric network family. While it is clear that SUR outperforms other approaches, one may raise a concern that the improvement is due to the increased number of parameters, that is, we use 8 times more parameters than in a single ResNet18. To address this concern, we use a parametric network family [34] that has only 0.5% more parameters than a single ResNet18. As described in Section 3.3, the parametric network family uses ResNet18 as a base feature extractor and FiLM [31] layers for feature modulation. The total number of additional parameters, represented by all domain-specific FiLM layers is approximately 0.5% of ResNet18 parameters. For comparison, CNAPs adaptation mechanism is larger than ResNet18 itself. To train the parametric network family, we first train a base CNN feature extractor on ImageNet. Then, for each remaining training dataset, we learn a set of FiLM layers, as detailed in Section 3.3. To obtain a multi-domain feature set for an image, we run inference 8 times, each time with a set of FiLM layers, corresponding to a different domain, as described in 3.3. Once the multi-domain feature is built, our selection mechanism is applied to it as described before (see Section 4.1). The results of using SUR with a parametric network family are presented in Table 2 as "SUR-pf". The table suggests that the accuracy on datasets similar to ImageNet is improved suggesting that parameter sharing is beneficial in this case and confirms the original findings of [34]. However, the opposite is true for different visual domains such as Fungi and QuickDraw. It implies that to do well on very different domains, the base CNN filters must be learned on those datasets from scratch, and feature modulation is not competitive.

Reducing the number of training domains. Another way to increase SUR's efficiency is to use fewer domains. This means training fewer domain-specific
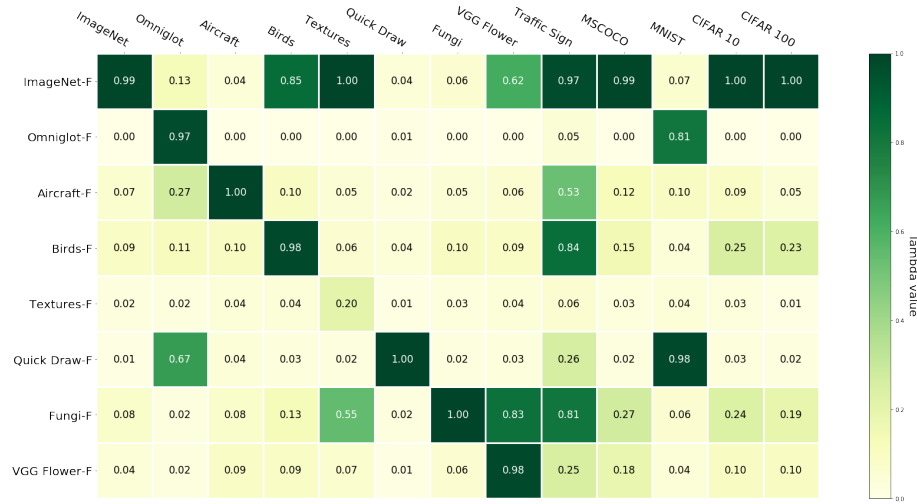
Fig. 4: **Frequency of selected features depending on the test domain in Meta-Dataset.** The top row indicates a testing dataset. The leftmost column a dataset the feature extractor has been trained on. A cells at location i, j reflects the average value of selection parameter λ_i assigned to the i-th feature extractor when tested on j-th dataset. The values are averaged over 600 few-shot test tasks for each dataset.

feature extractors, faster inference and selection. To achieve this goal, we merge similar datasets together into separate visual domains, and train a single feature extractor for each such domain. Here, we take 8 datasets within MetaDataset and use them to form 3 following visual domains: [ImageNet, Aircraft, CU-Birds, Textures, VGG-Flower], [Omniglot, Quick Draw], [Fungi, ImageNet, Aircraft]. Then, we train 3 feature extractors, one for each such domain. These 3 feature extractors constitute the multi-domain feature set that is then used for selection with SUR, as described in Section 4.1. Doing so leads to a 2x speed-up in training and inference. Moreover, as Table 2 suggests, using 'SUR-merge' allows us to achieve better performance. We attribute this to learning better representation thanks to the increased number of relevant training samples per feature extractor.

4.3   Analysis of Feature Selection

In this section, we analyze optimized selection parameters λ when applying SUR on MetaDataset. Specifically, we perform the experiments from Section 4.2, where we select appropriate representations from a set, generated by 8 independent networks. For each test dataset, we then average selection vectors λ (after being optimized for 40 SGD steps) over 600 test tasks and present them in Figure 4. First, we can see that the resulting λ are sparse, confirming that most of the time SUR actually select a few relevant features rather than takes all features with similar weights. Second, for a given test domain, SUR tends to select feature

Method	Aggregation	5-shot	1-shot
Cls	last	76.28 ±0.41	60.09 ±0.61
	concat	75.67 ±0.41	57.15 ±0.61
	SUR	**79.25** ±0.41	**60.79** ±0.62
DenseCls	last	78.25 ±0.43	62.61 ±0.61
	concat	79.59 ±0.42	62.74 ±0.61
	SUR	**80.04** ±0.41	**63.13** ±0.62
Robust20-dist	last	81.06 ±0.41	**64.14** ±0.62
	concat	80.79 ±0.41	63.22 ±0.63
	SUR	**81.19** ±0.41	63.93 ±0.63

Table 3: **Comparison to other methods on 1- and 5-shot *mini*-ImageNet.** The first column specifies a way of training a feature extractor, while the second column reflects how the final image representation is constructed. The two last columns display the accuracy on 1- and 5-shot learning tasks. The average is reported over 1 000 independent experiments with 95% confidence interval. The best accuracy is in bold.

extractors trained on similar visual domains. Interestingly, for datasets coming from exactly the same distribution, *i.e.* `CIFAR 10` and `CIFAR 100`, the averaged selection parameters are almost identical. All of the above suggests that the selection parameters could be interpreted as encoding the importance of features' visual domains for the test domain.

4.4   Single-domain Few-shot Classification

In this section, we show that our feature selection strategy can be effective for various problems, not just for multi-domain few-shot learning. As an example, we demonstrate the benefits of SUR on few-shot classification, when training and testing classes come from the same dataset. Specifically, we show how to use our selection strategy in order to improve existing adaptation-free methods.

To test SUR in the single-domain scenario we use *mini*-ImageNet benchmark and solve 1-shot and 5-shot classification, as described in Section 4.1. When only one domain is available, obtaining a truly multi-domain set of features is not possible. Instead, we construct a proxy for such a set by using activations of network's intermediate layers. In this approximation, each intermediate layer provides features corresponding to some domain. Since different layers extract different features, selecting the relevant features should help on new tasks.

We experiment with 3 adaptation-free methods. They all use the last layer of ResNet12 as image features and build a NCC on top, however, they differ in a way the feature extractor is trained. The method we call "Cls", simply trains ResNet12 for classification on the meta-training set. The work of [22] performs dense classification instead (dubbed "DenseCls"). Finally, the "Robust20-dist" feature extractor [7] is obtained by ensemble distillation. For any method, the "multi-domain" feature set is formed from activations of the last 6 layers of the network. This is because the remaining intermediate layers do not contain useful for the final task information, as we show in Appendix.

Here, we explore different ways of exploiting such "multi-domain" set of features

for few-shot classification and report the results in Table 3. We can see that using SUR to select appropriate for classification layers usually works better than using only the penultimate layer (dubbed "last") or concatenating all the features together (denoted as "concat"). For Robust20-dist, we observe only incremental improvements for 5-shot classification and negative improvement in 1-shot scenario. The penultimate layer in this network is probably the most useful for new problems and, if not selected, may hinder the final accuracy.

5   Acknowledgments

References

1. Bietti, A., Mialon, G., Chen, D., Mairal, J.: A kernel perspective for regularizing deep neural networks. In: International Conference on Machine Learning (ICML) (2019)
2. Bilen, H., Vedaldi, A.: Universal representations: The missing link between faces, text, planktons, and cat breeds. arXiv preprint arXiv:1701.07275 (2017)
3. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
4. Chen, W.Y., Liu, Y.C., Kira, Z., Wang, Y.C., Huang, J.B.: A closer look at few-shot classification. In: International Conference on Learning Representations (ICLR) (2019)
5. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
6. Doersch, C., Zisserman, A.: Multi-task self-supervised visual learning. In: Proceedings of the International Conference on Computer Vision (ICCV) (2017)
7. Dvornik, N., Schmid, C., Mairal, J.: Diversity with cooperation: Ensemble methods for few-shot classification. In: Proceedings of the International Conference on Computer Vision (ICCV) (2019)
8. Dvornik, N., Shmelkov, K., Mairal, J., Schmid, C.: Blitznet: A real-time deep network for scene understanding. In: Proceedings of the International Conference on Computer Vision (ICCV) (2017)
9. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning (ICML) (2017)
10. Garnelo, M., Rosenbaum, D., Maddison, C.J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y.W., Rezende, D.J., Eslami, S.: Conditional neural processes. arXiv preprint arXiv:1807.01613 (2018)
11. Gidaris, S., Bursuc, A., Komodakis, N., Pérez, P., Cord, M.: Boosting few-shot visual learning with self-supervision. In: Proceedings of the International Conference on Computer Vision (ICCV) (2019)

12. Gidaris, S., Komodakis, N.: Dynamic few-shot visual learning without forgetting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

13. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: International Conference on Learning Representations (ICLR) (2018)

14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)

15. Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., Igel, C.: Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In: International Joint Conference on Neural Networks (IJCNN) (2013)

16. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015), International Conference on Machine Learning (ICML)

17. Jonas Jongejan, Henry Rowley, T.K.J.K., Fox-Gieg, N.: Fine-grained visual classification of aircraft. quickdraw.withgoogle.com (2016)

18. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)

19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NeurIPS) (2012)

20. Lake, B.M., Salakhutdinov, R., Tenenbaum, J.B.: Human-level concept learning through probabilistic program induction. Science (2015)

21. LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D.: Backpropagation applied to handwritten zip code recognition. Neural computation (1989)

22. Lifchitz, Y., Avrithis, Y., Picard, S., Bursuc, A.: Dense classification and implanting for few-shot learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

23. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Proceedings of the European Conference on Computer Vision (ECCV) (2014)

24. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector. In: Proceedings of the European Conference on Computer Vision (ECCV) (2016)

25. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)

26. Mairal, J., Bach, F., Ponce, J., et al.: Sparse modeling for image and vision processing. Foundations and Trends® in Computer Graphics and Vision (2014)

27. Maji, S., Kannala, J., Rahtu, E., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. Tech. rep. (2013)

28. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Distance-based image classification: Generalizing to new classes at near-zero cost. IEEE transactions on Pattern Analysis and Machine Intelligence (PAMI) (2013)

29. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Sixth Indian Conference on Computer Vision, Graphics & Image Processing (2008)

30. Oreshkin, B., López, P.R., Lacoste, A.: Tadam: Task dependent adaptive metric for improved few-shot learning. In: Advances in Neural Information Processing Systems (NeurIPS) (2018)

31. Perez, E., Strub, F., De Vries, H., Dumoulin, V., Courville, A.: Film: Visual reasoning with a general conditioning layer. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

32. Qiao, S., Liu, C., Shen, W., Yuille, A.L.: Few-shot image recognition by predicting parameters from activations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

33. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning. In: International Conference on Learning Representations (ICLR) (2017)

34. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Efficient parametrization of multi-domain deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

35. Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. arXiv preprint arXiv:1803.00676 (2018)

36. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NeurIPS) (2015)

37. Requeima, J., Gordon, J., Bronskill, J., Nowozin, S., Turner, R.E.: Fast and flexible multi-task classification using conditional neural adaptive processes. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)

38. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)

39. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. Proceedings of the International Conference on Computer Vision (ICCV) (2015)

40. Rusu, A.A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., Hadsell, R.: Meta-learning with latent embedding optimization. In: Advances in Neural Information Processing Systems (NeurIPS) (2018)

41. Saikia, T., Brox, T., Schmid, C.: Optimized generic feature learning for few-shot classification across domains. arXiv preprint arXiv:2001.07926 (2020)

42. Schmidhuber, J., Zhao, J., Wiering, M.: Shifting inductive bias with success-story algorithm, adaptive levin search, and incremental self-improvement. Machine Learning (1997)

43. Schroeder, B., Cui, Y.: Fgvcx fungi classification challenge 2018. github.com/visipedia/fgvcx_fungi_comp (2018)

44. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems (NeurIPS) (2017)

45. Thrun, S.: Lifelong learning algorithms. In: Learning to learn, pp. 181–209 (1998)

46. Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.A., Larochelle, H.: Meta-dataset: A dataset of datasets for learning to learn from few examples. arXiv preprint arXiv:1903.03096 (2019)

47. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: Advances in Neural Information Processing Systems (NeurIPS) (2016)

48. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)

49. Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., Liu, W.: Cosface: Large margin cosine loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
50. Yann LeCun, C.C., Burges, C.: Mnist handwritten digit database. http://yann.lecun.com/exdb/mnist (2010)
51. Yoshida, Y., Miyato, T.: Spectral norm regularization for improving the generalizability of deep learning. arXiv preprint arXiv:1705.10941 (2017)
52. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems (NeurIPS) (2014)
53. Zeiler, M.D.: Adadelta: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
54. Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A.S., Neumann, M., Dosovitskiy, A., et al.: The visual task adaptation benchmark. arXiv preprint arXiv:1910.04867 (2019)

A   Implementation and Datasets details

A.1   Full MetaDataset Description

The MetaDataset includes `ImageNet` [39] (1000 categories of natural images), `Omniglot` [20] (1623 categories of black-and-white hand-written characters from different alphabets), `Aircraft` [27] (100 classes of aircraft types), `CU-Birds` [48] (200 different bird species), Describable `Textures` [5] (43 categories for textures), `Quick Draw` [17] (345 different categories of black-and-white sketches), `Fungi` [43] (1500 mushroom types), `VGG-Flower` [29] (102 flower species), `Traffic Sign` [15] (43 classes of traffic signs) and `MSCOCO` [23] (80 categories of day-to-day objects). For testing, we additionally employ `MNIST` [50] (10 hand-written digits) `CIFAR10` [18] (10 classes of common objects), and `CIFAR100` [18] (100 classes of common objects). Figure 5 illustrated random samples drawn from each dataset.

A.2   MetaDataset training details

When using multiple ResNet18 on MetaDataset (a single ResNet per dataset) to build a multi-domain representation, we train the networks according to the following procedure. For optimization, we use SGD with momentum and adjust the learning rate using cosine annealing [**?**]. The starting learning rate, the maximum number of training iterations ("Max iter.") and annealing frequency ("annealing freq.") are set individually for each dataset. To regularize training, we use data augmentation, such as random crops and random color augmentations, and set a constant weight decay of 7×10^{-4}. For each dataset, we run a grid search over batch size in [8, 16, 32, 64] and pick the one that maximizes accuracy on the validation set. The hyper-parameters maximizing the validation accuracy are given in Table 4.

When training a parametric network family for building multi-domain representations, we start by adopting a ResNet18 already trained on ImageNet, that we keep fixed for the rest of the training procedure. For each new dataset, we then train a set of domain-specific FiLM layers, modulating intermediate ResNet layers, as described in Section 3.3 of the original paper. Here, we also use cosine annealing as learning rate policy, employ weight decay and data augmentation, as specified above. In Table 5, we report the training hyper-parameters for each of the datasets.

A.3   mini-ImageNet training details

All the methods we evaluate on *mini*-ImageNet use ResNet12 [30] as a feature extractor. It is trained with batch size 200 for 48 epochs. For optimization, we use Adam optimizer [**?**] with initial learning rate 0.1 which is kept constant for the first 36 epochs. Between epochs 36 and 48, the learning rate was exponentially decreased from 0.1 to 10^{-5}, *i.e.* by dividing the learning rate by $10^{\frac{1}{3}}$ after each epoch. As regularization, we use weight decay with 5×10^{-4} multiplier and data augmentation such as random crops, flips and color transformations.

Test Dataset	learning rate	weight decay	Max iter.	annealing freq.	batch size
ImageNet	3×10^{-2}	7×10^{-4}	480,000	48,000	64
Omniglot	3×10^{-2}	7×10^{-4}	50,000	3,000	16
Aircraft	3×10^{-2}	7×10^{-4}	50,000	3,000	8
Birds	3×10^{-2}	7×10^{-4}	50,000	3,000	16
Textures	3×10^{-2}	7×10^{-4}	50,000	1,500	32
Quick Draw	1×10^{-2}	7×10^{-4}	480,000	48,000	64
Fungi	3×10^{-2}	7×10^{-4}	480,000	15,000	32
VGG Flower	3×10^{-2}	7×10^{-4}	50,000	1,500	8

Table 4: **Training hyper-parameters of individual feature networks on Meta-Dataset**. The first column indicates the dataset used for training. The first row gives the name of he hyper-parameter. The body of the table contains hyper-parameters that produced the most accurate model on the validation set.

Test Dataset	learning rate	weight decay	Max iter.	annealing freq.	batch size
Omniglot	3×10^{-2}	7×10^{-4}	40,000	3,000	16
Aircraft	1×10^{-2}	7×10^{-4}	30,000	1,500	32
Birds	3×10^{-2}	7×10^{-4}	30,000	1,500	16
Textures	3×10^{-2}	7×10^{-4}	40,000	1,500	16
Quick Draw	1×10^{-2}	7×10^{-4}	400,000	15,000	32
Fungi	1×10^{-2}	7×10^{-4}	400,000	15,000	32
VGG Flower	1×10^{-2}	7×10^{-4}	30,000	3,000	16

Table 5: **Training hyper-parameters of the parametric network family on MetaDataset**. The first column indicates the dataset used for training. The first row gives the name of he hyper-parameter. The body of the table contains hyper-parameters that produced the most accurate model on the validation set.

B   Additional Experiments and Ablation Study

B.1   Additional results on MetaDataset

Here we elaborate on using SUR with a multi-domain set of representations obtained from independent feature extractors (see Section 3.2), report an ablation study on varying the number of extractors in the multi-domain set, and report detailed results, corresponding to Figure 3 (a) of the original paper. Specifically, we use 8 domain-specific ResNet18 feature extractors to build a multi-domain representation and evaluate SUR against the baselines. The results are reported in Table 6, which corresponds to Figure 3 (a) of the original paper.

In the following experiment, we remove feature extractors trained on `Birds`, `Textures` and `VGG Flower` from the multi-domain feature set and test the performance of SUR on the set of remaining 5 feature extractors. We chose to remove these feature extractors as none of them gives the best performance on any of the test sets. Hence, they probably do not add new knowledge to the multi-domain set of features. The results are reported in Table 6 (a) as "SUR (5/8)". As we

Test Dataset	ImageNet-F	Union-F	Concat-F	SUR	SUR (5/8)
ImageNet	**56.3±1.0**	44.6±0.7	19.5±1.6	**56.0±1.2**	**56.1±1.1**
Omniglot	67.5±1.2	86.1±0.9	91.5±0.5	**93.0±0.4**	**93.1±0.4**
Aircraft	50.4±0.9	82.2±0.6	33.7±1.4	**85.7±0.3**	**85.5±0.4**
Birds	71.7±0.8	**72.1±1.1**	18.8±1.3	71.6±0.8	71.0±0.8
Textures	**70.2±0.7**	62.7±1.0	34.5±0.9	**70.3±0.9**	**70.4±0.9**
Quick Draw	52.3±1.0	70.7±0.9	51.2±0.9	**80.2±0.8**	**80.5±0.9**
Fungi	39.1±1.0	56.2±0.8	12.6±0.4	**62.8±1.1**	**63.1±1.0**
VGG Flower	**84.3±0.7**	82.5±0.8	40.3±1.2	83.6±0.8	83.3±0.8
Traffic Sign	63.1±0.8	63.8±0.9	48.2±0.6	**66.1±0.8**	63.6±0.9
MSCOCO	**52.8±1.0**	42.3±1.0	17.8±0.4	**52.4±1.1**	**52.8±1.1**
MNIST	77.2±0.7	84.8±0.6	89.6±0.7	91.2±0.5	**92.5±0.5**
CIFAR 10	**66.3±0.8**	51.4±0.8	34.7±0.8	64.6±0.9	65.8±0.9
CIFAR 100	**55.7±1.0**	39.5±1.0	18.9±0.6	54.5±1.0	**56.5±1.0**

Table 6: **Motivation for feature selection.** The table shows accuracy of different feature combinations on the Meta-Detaset test splits. The first column indicates the dataset the algorithms are tested on, the first row gives a name of a few-shot algorithm. The body of the table contains average accuracy and 95% confidence intervals computed over 600 few-shot tasks. The numbers in bold lie have intersecting confidence intervals with the most accurate method.

can see, selecting from the truncated set of features may be beneficial for some out-of-domain categories, which suggests that even the samples form of adaptation – selection – may overfit when very few samples are available. On the other hand, for a new dataset `Traffic Sign`, selecting from all features is beneficial. This result is not surprising, as one generally does not know what features will be useful for tasks not known beforehand, and thus removing seemingly useless features may result in a performance drop.

B.2   Analysis of Feature Selection on MetaDataset

Here, we repeat the experiment from Section 4.3, *i.e.* studying average values of selection parameters λ depending on the test dataset. Figure 6 reports the average selection parameters with corresponding confidence intervals. This is in contrast to Figure 4 of the original paper that reports the average values only, without confidence intervals.

B.3   Importance of Intermediate Layers on mini-ImageNet

We clarify the findings in Section 4.4 of the original paper and provide an ablation study on the importance of intermediate layers activations for the meta-testing performance. For all experiments on *mini*-ImageNet, we use ResNet12 as a feature extractor and construct a multi-domain feature set from activations of intermediate layers. In Table 7, we experiment with adding different layers outputs to the multi-domain set. The multi-domain set is then used to construct the final image representation either through concatenation "concat" or using

Method	1-3	4-6	7-9	10-12	Aggregation	5-shot	1-shot
					last	76.28 ±0.41	60.09 ±0.61
				✓	select	77.39 ±0.42	**61.02** ±**0.62**
Cls			✓	✓	select	**79.25** ±0.41	60.79 ±0.62
		✓	✓	✓	select	78.92 ±0.41	60.71 ±0.64
	✓	✓	✓	✓	select	78.80 ±0.43	60.55 ±0.62
				✓	concat	78.43 ±0.42	60.41 ±0.62
			✓	✓	concat	75.67 ±0.41	57.15 ±0.61
		✓	✓	✓	concat	70.90 ±0.40	53.53 ±0.61
	✓	✓	✓	✓	concat	69.40 ±0.40	51.21 ±0.60
					last	78.25 ±0.43	62.61 ±0.61
				✓	select	79.34 ±0.42	62.46 ±0.62
DenseCls			✓	✓	select	**80.04** ±0.41	**63.13** ±**0.62**
		✓	✓	✓	select	79.84 ±0.42	62.95 ±0.62
	✓	✓	✓	✓	select	79.49 ±0.43	62.58 ±0.63
				✓	concat	79.12 ±0.41	62.51 ±0.62
			✓	✓	concat	79.59 ±0.42	62.74 ±0.61
		✓	✓	✓	concat	77.63 ±0.42	60.14 ±0.61
	✓	✓	✓	✓	concat	76.07 ±0.41	57.78 ±0.61
					last	81.06 ±0.41	**64.14** ±**0.62**
				✓	select	**81.23** ±0.42	63.83 ±0.62
DivCoop			✓	✓	select	81.19 ±0.41	63.93 ±0.63
		✓	✓	✓	select	81.11 ±0.42	63.85 ±0.62
	✓	✓	✓	✓	select	81.08 ±0.42	63.71 ±0.62
				✓	concat	81.12 ±0.42	63.92 ±0.62
			✓	✓	concat	80.79 ±0.41	63.22 ±0.63
		✓	✓	✓	concat	80.52 ±0.42	62.48 ±0.61
	✓	✓	✓	✓	concat	80.36 ±0.42	61.30 ±0.61

Table 7: **Comparison to other methods on 1- and 5-shot *mini*-ImageNet.** The first column gives the name of the feature extractor. Columns 2-5 indicate if corresponding layers of ResNet12 were added to the multi-domain set of representations. Column "Aggregation" specifies how the multi-domain set was used to obtain a vector image representation. The two last columns display the accuracy on 1- and 5-shot learning tasks. To evaluate our methods we performed 1 000 independent experiments on *mini*-ImageNet-test and report the average and 95% confidence interval. The best accuracy is in bold.

SUR. The table suggests that adding the first 6 layers negatively influences the performance of the target task. While our SUR approach can still select relevant features from the full set of layers, the negative impact is especially pronounced for the "concat" baseline. This suggests that the first 6 layers do not contain useful for the test task information. For this reason, we do not include them in the multi-domain feature set, when reporting the results in Section 4.4.

We further provide analysis of selection coefficients assigned to different layers in Figure 7. We can see that for all methods, SUR picks from the last 6 layers most of the time. However, it can happen that some of the earlier layers are selected too. According to Table 7, these cases lead to a decrease in performance and suggest the SUR may overfit, when the number of samples if very low.

ImageNet

Omniglot
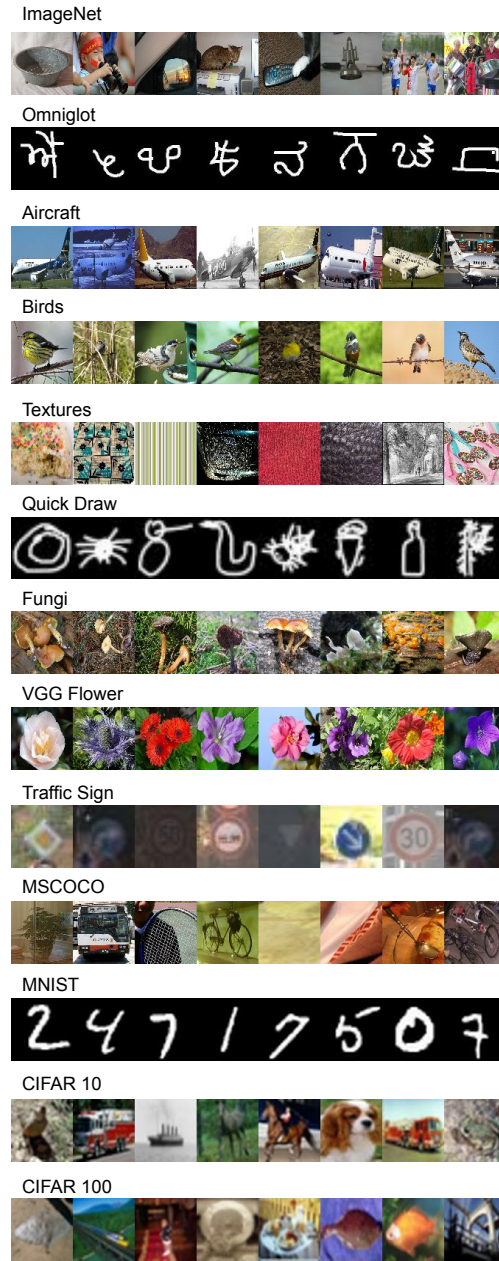

Aircraft

Birds

Textures

Quick Draw

Fungi

VGG Flower

Traffic Sign

MSCOCO

MNIST

CIFAR 10

CIFAR 100

Fig. 5: **Samples from all MetaDataset datasets** Each line gives 8 random samples from a dataset specified above.

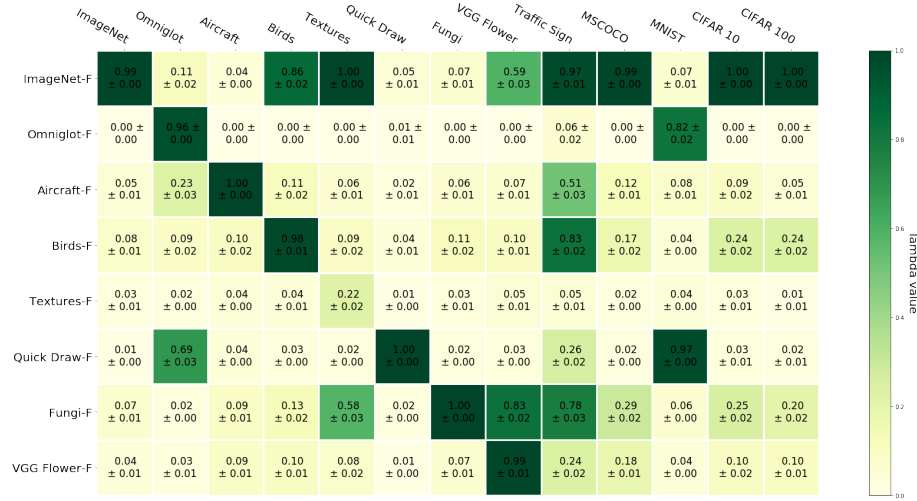|	ImageNet	Omniglot	Aircraft	Birds	Textures	Quick Draw	Fungi	VGG Flower	Traffic Sign	MSCOCO	MNIST	CIFAR 10	CIFAR 100
ImageNet-F	0.99 ± 0.00	0.11 ± 0.02	0.04 ± 0.00	0.86 ± 0.02	1.00 ± 0.00	0.05 ± 0.01	0.07 ± 0.01	0.59 ± 0.03	0.97 ± 0.01	0.99 ± 0.00	0.07 ± 0.01	1.00 ± 0.00	1.00 ± 0.00
Omniglot-F	0.00 ± 0.00	0.96 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.01	0.00 ± 0.00	0.00 ± 0.00	0.06 ± 0.02	0.00 ± 0.00	0.82 ± 0.02	0.00 ± 0.00	0.00 ± 0.00
Aircraft-F	0.05 ± 0.01	0.23 ± 0.03	1.00 ± 0.00	0.11 ± 0.02	0.06 ± 0.01	0.02 ± 0.01	0.06 ± 0.01	0.07 ± 0.01	0.51 ± 0.03	0.12 ± 0.01	0.08 ± 0.01	0.09 ± 0.02	0.05 ± 0.01
Birds-F	0.08 ± 0.01	0.09 ± 0.02	0.10 ± 0.02	0.98 ± 0.01	0.09 ± 0.02	0.04 ± 0.01	0.11 ± 0.02	0.10 ± 0.01	0.83 ± 0.02	0.17 ± 0.02	0.04 ± 0.00	0.24 ± 0.02	0.24 ± 0.02
Textures-F	0.03 ± 0.01	0.02 ± 0.00	0.04 ± 0.00	0.04 ± 0.01	0.22 ± 0.02	0.01 ± 0.00	0.03 ± 0.01	0.05 ± 0.01	0.05 ± 0.01	0.02 ± 0.00	0.04 ± 0.00	0.03 ± 0.01	0.01 ± 0.01
Quick Draw-F	0.01 ± 0.00	0.69 ± 0.03	0.04 ± 0.00	0.03 ± 0.00	0.02 ± 0.00	1.00 ± 0.00	0.02 ± 0.00	0.03 ± 0.00	0.26 ± 0.02	0.02 ± 0.00	0.97 ± 0.00	0.03 ± 0.01	0.02 ± 0.01
Fungi-F	0.07 ± 0.01	0.02 ± 0.00	0.09 ± 0.01	0.13 ± 0.02	0.58 ± 0.03	0.02 ± 0.00	1.00 ± 0.00	0.83 ± 0.02	0.78 ± 0.03	0.29 ± 0.02	0.06 ± 0.00	0.25 ± 0.02	0.20 ± 0.02
VGG Flower-F	0.04 ± 0.01	0.03 ± 0.01	0.09 ± 0.01	0.10 ± 0.01	0.08 ± 0.02	0.01 ± 0.00	0.07 ± 0.01	0.99 ± 0.01	0.24 ± 0.02	0.18 ± 0.01	0.04 ± 0.00	0.10 ± 0.02	0.10 ± 0.01

Fig. 6: **Frequency of selected features depending on the test domain in Meta-Dataset.** The top row indicates a testing dataset. The leftmost column presents a dataset the feature extractor has been trained on. A cells at location i, j reflects the average value of selection parameter λ_i assigned to the i-th feature extractor when tested on j-th dataset with corresponding 95% confidence intervals. The values are averaged over 600 few-shot test tasks for each dataset.

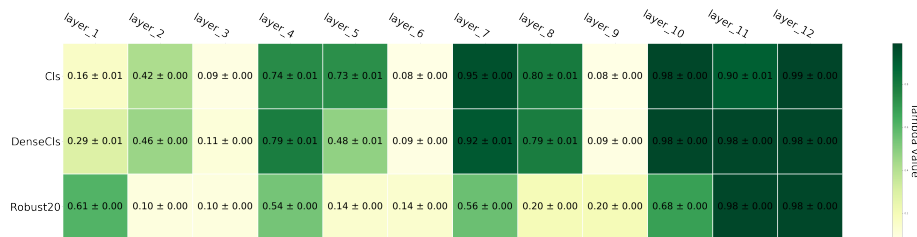|	layer_1	layer_2	layer_3	layer_4	layer_5	layer_6	layer_7	layer_8	layer_9	layer_10	layer_11	layer_12
Cls	0.16 ± 0.01	0.42 ± 0.00	0.09 ± 0.00	0.74 ± 0.01	0.73 ± 0.01	0.08 ± 0.00	0.95 ± 0.00	0.80 ± 0.01	0.08 ± 0.00	0.98 ± 0.00	0.90 ± 0.01	0.99 ± 0.00
DenseCls	0.29 ± 0.01	0.46 ± 0.00	0.11 ± 0.00	0.79 ± 0.01	0.48 ± 0.01	0.09 ± 0.00	0.92 ± 0.01	0.79 ± 0.01	0.09 ± 0.00	0.98 ± 0.00	0.98 ± 0.00	0.98 ± 0.00
Robust20	0.61 ± 0.00	0.10 ± 0.00	0.10 ± 0.00	0.54 ± 0.00	0.14 ± 0.00	0.14 ± 0.00	0.56 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.68 ± 0.00	0.98 ± 0.00	0.98 ± 0.00

Fig. 7: **Frequency of selecting intermediate layer's activation's on mini-ImageNet for 5-shot classification.** The top row indicates intemediate layer. The leftmost column gives the name of a method used to pre-train the feature extractor. Each cells reflects the average value of selection parameter λ_i assigned to the i-th intemediate layer with corresponding 95% confidence intervals. The values are averaged over 1000 few-shot test tasks for each dataset.