Pushing the Limits of Simple Pipelines for Few-Shot Learning: External Data and Fine-Tuning Make a Difference

Shell Xu Hu[1]   Da Li[1*]   Jan Stühmer[1*]   Minyoung Kim[1*]   Timothy M. Hospedales[1,2]

[1]Samsung AI Center Cambridge   [2]University of Edinburgh

{shell.hu, da.li1, jan.stuhmer, k.minyoung, t.hospedales}@samsung.com

Abstract

Few-shot learning (FSL) is an important and topical problem in computer vision that has motivated extensive research into numerous methods spanning from sophisticated meta-learning methods to simple transfer learning baselines. We seek to push the limits of a simple-but-effective pipeline for real-world few-shot image classification in practice. To this end, we explore few-shot learning from the perspective of neural architecture, as well as a three stage pipeline of pre-training on external data, meta-training with labelled few-shot tasks, and task-specific fine-tuning on unseen tasks. We investigate questions such as: ① How pre-training on external data benefits FSL? ② How state of the art transformer architectures can be exploited? and ③ How to best exploit fine-tuning? Ultimately, we show that a simple transformer-based pipeline yields surprisingly good performance on standard benchmarks such as Mini-ImageNet, CIFAR-FS, CDFSL and Meta-Dataset. Our code is available at https://hushell.github.io/pmf.

1. Introduction

Mainstream supervised deep learning achieves excellent results in applications where huge annotated datasets are available. However, this assumption is not met in many applications where data (e.g., rare categories), or the cost of human annotation are prohibitive bottlenecks. This has motivated a large and growing set of research in few-shot learning (FSL), which aims to emulate the human ability to learn new concepts from few training examples. The FSL challenge has proven fertile ground for developing and testing a vast array of sophisticated research ideas spanning metric learning [55, 57], gradient-based meta-learning [29], program induction [40], differentiable optimization layers [41], hypernetworks [9], neural optimizers [50], transductive label propagation [51], neural loss learning [4], Bayesian neural priors [66] and more [64]. But how much practical progress
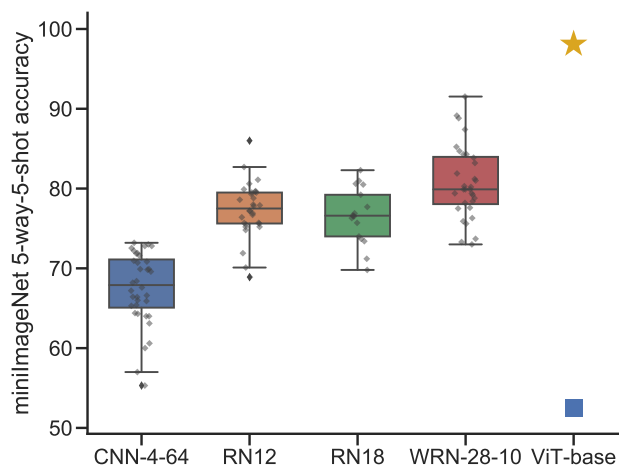
*Equal contributions.

Figure 1. **How does pre-training and architecture affect few-shot learning?** Learning from a few shots can be achieved by a) meta-learning [62,66] and b) transfer learning from self-supervised foundation models pre-trained on large-scale external data [18,49]. While the majority of FSL community focuses on the former, we show that the latter can be more effective because it enables the use of stronger architectures such as vision transformer (ViT) [25] – and can be combined with simple meta-learners such as ProtoNet. The figure shows results aggregated from dozens of studies from the past 5 years of FSL research and the result of ProtoNet + ViT backbone + contrastive language-image pretraining (CLIP) [49] (yellow star). To emphasize the importance of pre-training, ProtoNet + randomly initialized ViT (blue square) is also compared.

have we made based on all these technical advances?

A few studies [19, 20, 23, 47, 59, 63] have investigated whether simpler baselines can offer comparable performance to sophisticated state of the art few-shot learners. While there is no conclusive answer, due to on-going developments in both sophisticated learners [66] and simple baselines, there is a trend that simple approaches often perform surprisingly well compared to sophisticated counterparts. Their simplicity and efficacy leads these simple methods to be taken up in many practical applications of few-shot learning from

medical data analysis [11] to electronic engineering [39].

We follow this line of enquiry, but go further in investigating previously under-studied factors that influence the performance of simple few-shot pipelines. In particular we start with a simple ProtoNet [55]-like pipeline, and investigate three practically important design choices: pre-training data, neural architecture, and fine-tuning in meta-test.

Source data   While FSL addresses the small data regime, in reality FSL research is almost always about algorithms to transfer knowledge from large scale source tasks (aka meta-train) to small scale target tasks (aka meta-test). Existing literature almost always controls the source data, in order to carefully compare the impact of different knowledge transfer mechanisms of interest from hyper-networks [9] to gradient-based meta-learners [29]. While this is helpful to drive research on sophisticated *algorithms*, it does not answer the question of how choice of source *data* impacts performance? This question has been studied in other areas of vision and pattern recognition [10, 31, 56], but not for FSL. This is unhelpful for consumers of computer vision FSL research, who would be interested to know how much a simple change of source data can improve their applications? Especially since freely available large datasets already exist [21, 58], and exploiting more external source data is easier in practice than implementing sophisticated state-of-the-art meta-learners. To this end we investigate the impact of unsupervised pre-training on external data – a workflow recently termed as exploiting a *foundation model* [10] – on FSL tasks. This small change has substantial impact compared to 5 years of FSL research (Figure 1). Although this may violate definitions of the FSL problem that strictly prescribe the source set, the efficacy of the approach may prompt reflection on whether this is the best problem definition to focus on.

Neural architecture   Similarly to the situation with source data, FSL studies often control neural architecture to a handful of small networks such as CNN-4-64 and ResNet-12. This is partly to enable fair comparison of FSL algorithms, but this particular suite of networks is also a consequence of the small size of the source datasets used for training in common benchmarks such as miniImageNet. Thus the architectures commonly studied in FSL are somewhat out-of-date with regard to state-of-the-art computer vision. We therefore ask to what extent state-of-the-art architectures such as vision transformers [25] can benefit few-shot performance, especially in conjunction with larger pre-training datasets?

Fine-tuning   The many studies in the FSL literature are somewhat divided in whether they advocate [29,50,61] some kind of fine-tuning during model deployment (aka meta-test) for individual tasks, or whether a fixed feature representation should be sufficient [41, 55, 63]. We also investigate this issue, and suggest that *fine-tuning is necessary for deploying foundation models to out-of-distribution tasks*. We also introduce an algorithmic improvement to fine-tuning by
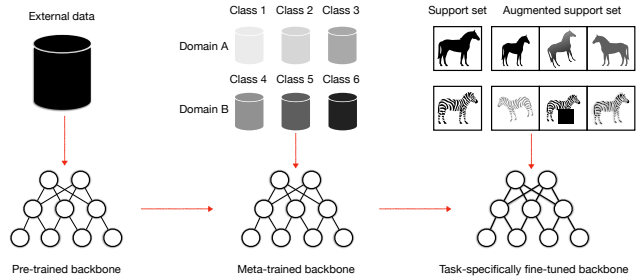

Figure 2. **Overview –** A schematic of the simple pre-training, meta-training, fine-tuning pipeline that we consider. Following the red arrows, the pipeline turns a pre-trained feature backbone into a task-specific one.

automating the learning rate selection via validation, which leads to a more performant pipeline for cross-domain FSL.

In summary, we advance few-shot learning by studying design choices of a simple pipeline [55] (Figure 2), rather than developing new algorithms. We answer questions including: *How does pre-training impact FSL? Can recent transformer architectures be adapted to FSL?* and *How to best exploit fine-tuning?* Based on this analysis we demonstrate a new baseline for FSL that surpasses state-of-the-art performance, while being simple and easy to implement.

2. Related Work

Few-shot learning   Few-shot learning is now a deep and widely studied area too large to review in detail here, and we refer to relevant surveys for an overview [34, 64]. A key point is that, despite the name, almost all FSL methods provide algorithms for transferring knowledge from a large set of source data, to a set of sparsely annotated target categories of interest. Much activity in the field falls under the umbrella of meta-learning [34], which aims to construct a data-efficient learner from the source (aka meta-train) dataset by simulating few-shot learning problems, and then deploy the customized learner on the target (aka meta-test) set. The resulting learner may take the form of an initialization [29], learned metric [55], Bayesian prior [66], or optimizer [50].

Simple-but-effective baselines   In competition with the plethora of sophisticated few-shot learners [34,64] such as those mentioned above, a number of recent studies have advocated strong baselines that perform comparably well while being simpler. These are often based on a transfer learning [65] pipeline. They apply a conventional deep learner on the source data, before adapting to the few-shot target data by training a simple linear [19, 47, 59] or centroid [63] classifier on the fixed representation, or fine-tuning the feature backbone as well [23]. These methods mostly use standardized FSL source datasets (such as miniImageNet) and architectures (such as ResNet-12 and WRN-10-28) to enable direct

comparisons of the advocated simple baselines to sophisticated learners. In contrast, we specifically aim to explore how far practical FSL performance can be pushed by exploiting other available pre-training datasets and architectures.

A few studies have evaluated FSL on a larger scale using datasets such as ImageNet1K [20] or ImageNet21K [23]. However by changing both the source and target sets, this does not make it clear how choice/scale of source data impacts a given target problem – the question that we answer here. Others have explored the impact of conventional pre-training prior to meta-learning [20] or as a regularizer during meta-learning [30] – but without exploiting extra data.

Bigger data and architectures The impact of source datasets is widely studied in standard supervised [56] and self-supervised [10, 31] learning in vision, and in pattern recognition applications outside of vision [3,10,13,22]. However, it is not widely evaluated in FSL, which is a surprising omission, since as we shall see it may well be the easiest way to improve practical FSL performance. Similarly, existing FSL methods are almost exclusively based on a few less common architectures (e.g., Conv-4-64 and ResNet-12), which maybe due to the very first experimental setup on small datasets like Omniglot [29, 62]. Transformers have seen limited use in FSL, mainly for metric learning [24], but not for feature extraction. We explore how recent transformer feature extractors can be trained and applied to FSL, especially when combined with a foundation model [10] pre-trained on larger source datasets.

Self-supervised & few-shot Our pipeline extends the typical unsupervised pre-train → supervised fine-tune workflow of the self-supervised research community [28, 38], which has recently demonstrated strong performance for low-shot supervised learning [15, 18, 27]. However, there has been limited direct comparison of self-supervised (SSL) and FSL community methods for data efficient learning due to different typical evaluation practices and benchmarks. For example, many SSL evaluations perform unsupervised representation learning on ImageNet, before performing few-shot supervised learning within ImageNet [15,18], which violates usual FSL community requirement of disjoint source and target data. One contribution of this paper is to provide a degree of comparison between and combination of the SSL and FSL approaches. For example, our MetaDataset, CDFSL and Teaser Figure 1 results, use disjoint source and target data but benefit from external self-supervised pre-training.

Cross-domain few-shot A FSL variant of particular practical interest is cross-domain few-shot [32], where the source/meta-train dataset is significantly different to the target/meta-test dataset. This is more challenging than the standard within-domain setting, but more practically relevant. This is because in many scenarios where FSL is of interest such as medical or earth observation imaging [32], the target

data for FSL is significantly different to available source data (such as (mini)ImageNet [21]). Major benchmarks of this type are CDFSL [32] and meta-dataset [61].

3. A Simple Pipeline for FSL

Problem formulation Few-shot learning (FSL) aims to learn a model with only a few annotated examples. One widely adopted formulation for FSL was introduced by Vinyals et al. [62] from a meta-learning perspective, where the assumption is that one should learn to solve new few-shot tasks based on previously seen experience of many similar few-shot tasks. Therefore, the FSL problem is usually organized in two phases: *meta-training* a few-shot learner on a distribution of training tasks and *meta-testing* the resulting learner by evaluating it on novel few-shot tasks. Within each phase, data arrives in an episodic fashion, where the "train-set" and "test-set" of each task are called *support set* and *query set* respectively to avoid terminology confusion. In the case of classification, the difficulty level of an episode is described as *K-way-N-shot*, which corresponds to learning a classifier for K classes given N examples per class in the support set. It is common to learn one model for each difficulty level, but a more realistic setting [61] is to learn a global model for various K's and N's. This is sometimes called *various-way-various-shot*, and we address this more practical setting here. This is also a reason to prefer simple pipelines over sophisticated meta-learners that may not be easily extended to the various-way-various-shot setting.

A different approach to small-data learning appears in the transfer learning [12,65] and self-supervision [10, 17] literature. In this case one pre-trains a model using some large source data, and then re-purposes it for the sparse data target task of interest. The pre-training step aims to reduce the sample complexity of learning the target problem in the adaptation step.

Although typically studied separately, both families of approach provide mechanisms for knowledge transfer from source data to the target few-shot problem of interest. Towards the goal of high performance few-shot learning, we combine both pre-training (typically on auxiliary unlabeled data, which is freely and ubiquitously available) and meta-learning (episodic training with labels) together in a simple sequential pipeline using a single feature extractor backbone. Our pipeline consists of three phases: 1) **pre-training** the feature backbone on unlabeled external data using self-supervised loss, 2) **meta-training** the feature backbone on labeled simulated few-shot tasks using ProtoNet [55] loss, and 3) deploying the feature backbone on novel few-shot tasks with optional **fine-tuning** on the augmented support set of each task. A schematic of our pipeline is shown in Figure 2, which we call P>M>F (i.e., the pipeline Pre-train → Meta-Learn → Fine-Tune). We next outline how the feature

backbone is updated in different stages.

3.1. Pre-training of backbone

We consider the feature backbones of ResNet [33] or ViT [25], to provide the foundation models in our pipeline. There are then several well-established self-supervised learning algorithms for the pre-training step: DINO [15] uses ImageNet1K and exploits the consistency in prediction between a large crop and multiple local crops of the same image, where a large crop is highly likely to overlap with a foreground object in the case of ImageNet images; BEiT [6] amounts to solving a masked image reconstruction task on the ImageNet-21K dataset in line with the original BERT pre-training [22] for text data; and CLIP [49] leverages image captions in the YFCC100m dataset to align image and caption representations in a common feature space. For more flexible architectures like ViT [25], pre-training on external data is important, as they are hard to train on common small-sized FSL benchmarks (Figure 1 and Table 1).

3.2. Meta-training with ProtoNet

As the goal is to build a simple pipeline, we consider the prototypical network (ProtoNet) [55], which constructs class centroids dynamically for each episode and then performs nearest centroid classification. Specifically, ProtoNet only requires a feature backbone f to map data points to a m-dimensional feature space: $f \colon \mathcal{X} \to \mathbb{R}^m$, and the probability of a query image x belonging to class k is given by

$$p(y = k|x) = \frac{\exp\left(-d(f(x), c_k)\right)}{\sum_{k'} \exp\left(-d(f(x), c_{k'})\right)}, \qquad (1)$$

where d is implemented by a cosine distance in our work as opposed to the commonly chosen Euclidean distance and c_k is the prototype of class k, defined as $c_k = \frac{1}{N_k} \sum_{i:y_i=k} f(x_i)$ and $N_k = \sum_{i:y_i=k} 1$ on the support set. Note that the prototypes can be computed regardless of the value of k. This enables ProtoNet to be trained and deployed under various-way-various-shot setting.

3.3. Meta-testing with fine-tuning

To be consistent with meta-training, by default, we deploy the meta-trained ProtoNet directly on all novel tasks. However, if the a novel task is drawn from an unseen domain, the learned feature representation may fail to generalize due to a substantial shift in the data distribution. To this end, we propose to fine-tune the feature backbone by a few gradient steps with the assistance of data augmentation. The details are summarized as PyTorch pseudo code in Algorithm 1.

Our fine-tuning algorithm is similar to that of [32,42] who fine-tune the model weights using the support set since this is the only accessible labeled data at meta-test time. We exploit the support set slightly differently: we use data augmentation

---

Algorithm 1 PyTorch pseudo code for fine-tuning

```
# Inputs: a task including supp_x, supp_y, query_x
# backbone_state: meta-trained backbone weights
# optimizer: Adam optimizer
# Outputs: logits

backbone = create_model_from_checkpoint(backbone_state)

def single_step(z):
    supp_f = backbone(supp_x)
    proto = compute_prototypes(supp_f, supp_y)
    f = backbone(z)
    logits = f.norm() @ proto.norm().T # cos similarity
    loss = cross_entropy_loss(logits, supp_y)
    return logits, loss

# fine-tuning loop
for i in range(num_steps):
    aug_supp_x = rand_data_augment(supp_x)
    _, loss = single_step(aug_supp_x)
    loss.backward() # back-prop
    optimizer.step() # gradient descent

logits, _ = single_step(query_x) # classification
```

---

to create a pseudo query set derived from the support set; as such, we do not need to compute prototypes using the support set and then again apply the prototypes on the same support set using eq. (1). Empirically, we find that updating the entire backbone is not a problem even though only a few examples are available for fine-tuning. While some layers could be frozen to save some computation, which layers to freeze may be domain-specific, complicating tuning.

Learning Rate Selection Fine-tuning performance is relatively sensitive to the choice of learning rate. To this end we propose an algorithm for automated learning rate tuning. We generate another pseudo query set from the support set using data augmentation, which is used to decide the learning rate within $\{0, 0.01, 0.001, 0.0001\}$. This adds $4\times$ computation cost to the model deployment, to perform per-episode learning rate selection. If multiple episodes can be used – essentially corresponding to assuming a per-domain set of validation episodes – learning rate can be selected per-dataset for greater efficiently. Details on this and other ablation studies can be found in the supplemental material.

4. Experiments

Meta-Training Datasets We use standard benchmarks to evaluate our proposed pipeline. **miniImageNet** [62] contains 100 classes from ImageNet-1k, which is then split into 64 training, 16 validation and 20 testing classes; each image is downsampled to 84×84. **CIFAR-FS** [8] is created by dividing the original CIFAR-100 into 64 training, 16 validation and 20 testing classes. The images are of size 32×32. **Meta-Dataset** [61] subsumes 10 public image datasets of a diverse range of domains: ImageNet-1k, Omniglot, FGVC-Aircraft, CUB-200-2011, Describable Textures, QuickDraw, FGVCx Fungi, VGG Flower, Traffic Signs and MSCOCO. Each dataset has train/val/test splits. We follow the two

training protocols proposed by [61] and [24] respectively. For the former, the train/val splits of the first 8 datasets (in-domain) are used for meta-training and validation, and the test splits of all datasets are used for meta-testing. The latter considers only ImageNet-1k's train-split for meta-training, and the other settings remain the same. For more details on Meta-Dataset we refer the readers to Appendix.3 of [61].

Evaluation   For evaluating few-shot classification performance, we simulate 600 episodes/tasks from the test-split for each dataset of interest. The evaluation metric is the average classification accuracy over tasks. For miniImageNet and CIFAR-FS, the convention is to evaluate 5-way-1-shot (5w1s) and 5-way-5-shot episodes, and the size of the query set for each episode is fixed to 15×5. For Meta-Dataset, the number of ways, shots and query images are sampled uniformly at random with respect to the dataset specifications, except for ImageNet-1k and Omniglot (they have specific sampling strategies according to the hierarchy of classes). In addition, we evaluate the (5w5s) meta-trained model from miniImageNet for a cross-domain evaluation (**CDFSL**) [32], where 4 out-of-domain datasets are considered, and the results are reported under 5-way-5/20/50-shot settings.

Training details   To avoid over-engineering training for different datasets and architectures, we adopt a common training strategy for meta-training the backbone from pre-trained model checkpoints (for both ResNet and ViT). This may lead to sub-optimal results for some cases, but it simplifies comparison. Specifically, we train the backbone for 100 epochs, where each epoch consists of 2000 episodes/tasks. We use a warm-up plus cosine annealing learning rate schedule: the learning rate starts from 10^{-6}, increases to 5×10^{-5} in 5 epochs and then gradually decreases to 10^{-6} with a cosine annealing. We use the validation set to decide when to early stop, and turn off strong regularization and data augmentation techniques for simplicity.

4.1. Analysis

We now use the pipeline outlined in Sec 3 to answer a series of questions about few-shot learner pipeline design. Notably, ① *How does pre-training regime affect FSL?* ② *Can contemporary architectures such as ViT be adapted to FSL?* ③ *How to exploit fine-tuning in meta-testing?*

4.1.1   Pre-training and architectures

We first evaluate the impact of pre-training regime (including algorithm and dataset), as well as neural architecture on FSL benchmarks Meta-Dataset [61] (train on 8 datasets), miniImageNet [62], and CIFAR-FS [8]. To clearly convey the configuration of each experiment, results in Table 1 are organized by architecture, pre-training algorithm (and dataset) and meta-training algorithm. We assume ProtoNet (nearest-centroid) classifier as the standard approach for

meta-testing throughout, and compare either episodically trained ProtoNet or nothing as the meta-learning step between pre-training and meta-testing (column MetaTr).

① **How does pre-training regime affect FSL?**   From the results in Table 1 we can draw the following conclusions: (i) Pre-training on ImageNet1K generally provides a significant improvement across the board compared to the conventional pipeline used by prior work which does not make use of pre-training (compare model M9 with M7 and M8, etc). (ii) We are primarily interested in unsupervised pre-training, with supervised pre-training being included as an unfair upper bound. However, state of the art unsupervised pre-training with DINO performs close to supervised pre-training (compare M3 vs M2, etc). This is noteworthy, because while there is some semantic overlap between some of the source (ImageNet1K) and target (Meta-Dataset, miniImageNet, CIFAR) datasets considered here, good performance can be achieved without using source *labels*, where there is no train-test label leakage[1]. (iii) Given a strong pre-training regime such as DINO, simple nearest centroid classification based on pre-trained features performs well (top block including M2, etc). In particular, off-the-shelf features from a foundation model without dataset-specific meta-learning perform favorably compared to conventional dataset-specific training of ProtoNet-ResNet18 (M2 vs M10), which is arguably the closest to industry standard in FSL. (iv) Nevertheless, dataset specific meta-learning does improve further (M7 vs M2, etc). Simple linear readout of a frozen foundation model [18, 27] is not competitive.

② **Can state of the art architectures such as ViT be adapted to FSL?**   Using the results in Table 1, we can also answer this question. In particular, while ViT does not train well on the smaller meta-train benchmarks (miniImageNet, CIFAR) compared to smaller architectures (see M6 vs M9, M10), it generally performs excellently when benefiting from large pre-training data (M6 vs M4). Overall ViT outperforms the industry standard ResNet18, as well as our ResNet50 baseline, across the board when benefitting from pre-training. We remark that our ResNet50 baseline also performs comparitively poorly without pre-training, especially on the smaller miniImageNet and CIFAR, suggesting that it is also too large to train well on the target datasets alone.

Other Foundation Models   Overall we can see that larger

---

[1]In the case of miniImageNet and Meta-Dataset, parts of ImageNet1K are used in both meta-train and meta-test splits. EG: since Meta-Dataset's ImageNet is uses a 712/288 source/target class split, this means that for one of Meta-Dataset's 10 domains, there is some data (but not label) overlap between pre-train and meta-test for some foundation models. As discussed in Sec. 2, this overlap is ubiquitious in typical self-supervision evaluation pipelines [15, 17]. It is less common in FSL evaluation pipelines, but corresponds to making a semi-supervised or transductive assumption in terms of data access as per [37, 44, 46, 51]. Nevertheless, we do not think this is a significant factor in the strong results, as CLIP's YFCC does not have this overlap and performs similarly to the ImageNet1K based models.

	Training Configuration			Benchmark Results		
ID	Arch	Pre Train	MetaTr	MD	miniIN	CIFAR
0	ViT-small	DINO (IN1K)	-	67.4	97.0	79.8
1	ViT-small	DeiT (IN1K)	-	67.5	98.8	84.6
2	ResNet50	DINO (IN1K)	-	63.8	91.5	76.1
3	ResNet50	Sup. (IN1K)	-	62.4	96.4	82.3
4	ViT-small	DINO (IN1K)	PN	78.4	98.0	92.5
5	ViT-small	DEIT (IN1K)	PN	79.3	99.4	93.6
6	ViT-small	-	PN	52.8	49.1	59.8
7	ResNet50	DINO (IN1K)	PN	72.4	92.0	84.0
8	ResNet50	Sup. (IN1K)	PN	70.2	97.4	87.6
9	ResNet50	-	PN	62.9	72.2	68.4
10	ResNet18	-	PN	63.3	73.7	70.2
11	ViT-base	DINO (IN1K)	PN	79.2	98.4	92.2
12	ViT-base	CLIP (YFCC)	PN	80.0	98.1	93.2
13	ViT-base	Sup (IN21K)	PN	81.4	99.2	96.7
14	ViT-base	BEIT (IN21K)	PN	82.8	99.0	97.5
15	ResNet50	CLIP (YFCC)	PN	75.0	92.2	82.6

Table 1. The impact of architecture and pre-training algorithm (dataset) on downstream few-shot learning performance on Meta-Dataset (MD), miniImageNet (miniIN) and CIFAR-FS. Meta-Dataset results are averaged over all target datasets while minIN and CIFAR results are 5-way-5-shot. ProtoNet (PN) nearest-centroid classifier is used throughout for few-shot learning on the support set during meta-test. MetaTr indicates the algorithm used for episodic learning on the corresponding benchmark.

	Train Config			Benchmark			
				miniIN		CIFAR	
ID	Arch	Pre Train	MetaTr	5/1	5/5	5/1	5/5
0	ViT-small	DINO (IN1K)	-	88.8	97.0	59.1	79.8
1	ViT-small	DINO (IN1K)	PN	93.1	98.0	81.1	92.5
2	ResNet18	-	MetaQDA	65.1	81.0	-	-
3	ViT-small	DINO (IN1K)	MetaQDA	92.0	97.0	77.2	90.1
4	ResNet12	-	MetaOptNet	64.1	80.0	72.8	85.0
5	ViT-small	DINO (IN1K)	MetaOptNet	92.2	97.8	70.2	84.1

Table 2. Impact of architecture and pre-training on state of the art few-shot learners

(such as ProtoNet, [55], MAML [29], MetaOptNet [41], etc), with those from the self-supervised community (such as DINO [15], SimCLR [17, 18], etc). This is partly because the popular evaluation protocol is different: For example 5-way-1-shot regime is popular the FSL community, vs 1% labels ($\approx$ 1000-way-10-shot in the case of ImageNet) in the SSL community; network architectures differ ($\leq$ResNet18 vs $\geq$ResNet50 respectively); and image resolutions differ ($84\times$ vs full). Our results provide a taster of such a direct comparison. Overall they suggest that frozen self-supervised foundation models (using extra pre-training data) are competitive out of the box compared to standard few-shot learners (using only meta-training data). However, more interestingly, combining these two paradigms as we have done, easily leads to state of the art performance on typical FSL metrics.

4.1.2 Fine-tuning

The previous experiments used a fixed feature extractor together with ProtoNet for meta-testing. We next investigate use of fine-tuning during meta-testing to further improve performance. We focus on the DINO pre-trained ViT models, based on their strong performance in Section 4.1.1.

③ **How to best exploit fine-tuning for meta-testing?** To answer this question, we compare vanilla feature transfer as explored previously, with ProtoNet, and ProtoNet with episode-wise fine-tuning on the support set (ProtoNet+FT) as outlined in Section 3.3. We use Meta-Dataset including both conditions of treating ImageNet alone as the source, and joint meta-training on all of Meta-Dataset. From the results in Figure 3 and Table 3 we can draw the following conclusions: (i) Meta-training on the full Meta-Dataset improves on meta-training on ImageNet-training alone (M5 vs M1). (ii) Fine-tuning during meta-test improves substantially in the out-of-distribution datasets, and especially in the case where meta-training is conducted on ImageNet, and then deployed across-domain to all the other Meta-Dataset tasks: See Out-D column and M2 vs M1 in Table 3; blue vs orange bars in Figure 3 for OmniGlot, QuickDraw, traffic signs, etc. However, for the condition where more Meta-Dataset domains are used for training and testing, fine-tuning has inconsistent impact across domains: While it is helpful for

pre-training data sources, and recent architectures make a huge difference to downstream FSL performance on standard benchmarks. We also compared a selection of other foundation models [10] in M11-15. We can see that (i) All the foundation models lead to substantial improvements on standard within-dataset training (M10,M9), (ii) The largest foundation models using, e.g., ViT-base and ImageNet21K or YFCC data source lead to strongest performance across the board, but do not outperform hugely the more economic DINO+ImageNet1K-based ViT-small (M4). For efficiency of pre-training and deployment, we take this to be our default model in the following section.

①+② **How does pre-training and architecture impact other Few-Shot Learners?** Our main experiments built upon ProtoNet as a widely used industry standard. We next explore how our pipeline impacts two few-shot learners that are more representative of recent state of the art, namely MetaOptNet [41] and MetaQDA [66]. From the results in Table 2, we can see that: (i) MetaQDA and MetaOptNet do improve on direct feature transfer (M5 and M3 vs M0) and on the simpler ResNet features they were initially evaluated with (M5 vs M4, M3 vs M2). But (ii) With the stronger features, they are outperformed by the simpler ProtoNet learner (M3 and M5 vs M1). This suggests previous conclusions about comparative meta-learner performance may need re-evaluating in this new regime of stronger features.

Discussion Existing literature generally fails to directly compare algorithms from the few-shot learning community

M	Arch	PreTr	MetaTr	MetaTe	Avg	Out-D
1	ViT-small	DINO	PN (IN)	PN	68.38	67.68
2	ViT-small	DINO	PN (IN)	PN+FT(lr=0.01)	76.05	76.54
3	ViT-small	DINO	PN (IN)	PN+FT(lr=0.001)	74.47	74.51
4	ViT-small	DINO	PN (IN)	PN+FT(Tuned)	77.53	77.85
5	ViT-small	DINO	PN (MD)	PN	78.43	55.71
6	ViT-small	DINO	PN (MD)	PN+FT(lr=0.01)	76.09	73.26
7	ViT-small	DINO	PN (MD)	PN+FT(lr=0.001)	74.64	69.97
8	ViT-small	DINO	PN (MD)	PN+FT(Tuned)	83.13	75.72

Table 3. Fine-tuning (FT) during meta-test on Meta-Dataset. The meta-train (MetaTr) setting indicates the source dataset as ImageNet only (IN) or full MetaDataset (MD). Results are the averages across all domains within meta-dataset (Avg), and just the out-of-distribution subset (Out-D).

the remaining OOD datasets, it is not helpful overall (M5 vs M6 for Avg and Out-D). Overall feature backbone updates by fine-tuning are more helpful for domains unseen during meta-training, concurring with [42, 61]. On analysing the inconsistent impact of fine-tuning, we found this is due to difficulty in choosing an appropriate learning rate. Using any single learning rate throughout, as we did above (lr=0.01) is poorly tuned for some datasets. We therefore also explore our learning rate selection heuristic proposed in Section 3.3, and we see this leads to the best performance (M4 vs M2).
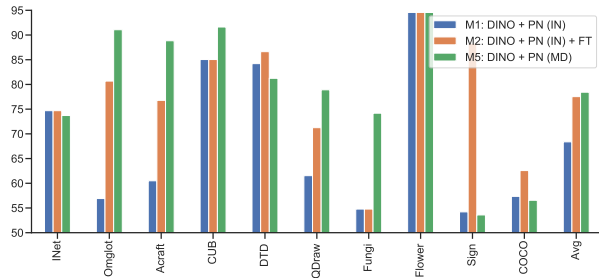

Figure 3. The impact of fine-tuning during meta-test on Meta-Dataset. Held out datasets such as Signs and COCO benefit from fine-tuning; as do those very different from ImageNet such as omniglot and QuickDraw.

4.2. Results on standard benchmarks

We call our pipeline **P>M>F**, which can be instantiated with any pre-training algorithm and backbone architectures, e.g., DINO > ProtoNet (PN) > Fine-tuning (FT). We next compare our pipeline with prior state of the art. We emphasize that our results are not directly comparable to much prior SotA in terms of architecture and use of external data. We draw this comparison to see how simple changes (such as exploiting publicly available pre-training data and updating to a modern architecture) compares against 5 years of intensive research on FSL algorithms. The results for single-domain miniImageNet and CIFAR-FS are summarized in Table 4,

Method (Backbone)	Ext. dat.	Ext. lab.	CIFAR-FS		MiniImageNet	
			5w1s	5w5s	5w1s	5w5s
Inductive						
ProtoNet (CNN-4-64) [55]			49.4	68.2	55.5	72.0
Baseline++ (CNN-4-64) [19]					48.2	66.4
MetaOpt-SVM (ResNet12) [41]			72.0	84.3	61.4	77.9
Meta-Baseline (ResNet12) [20]					68.6	83.7
RS-FSL (ResNet12) [2]		✓			65.3	
Transductive						
Fine-tuning (WRN-28-10) [23]			76.6	85.8	65.7	78.4
SIB (WRN-28-10) [35]			80.0	85.3	70.0	79.2
PT-MAP (WRN-28-10) [36]			**87.7**	90.7	82.9	88.8
CNAPS + FETI (ResNet18) [7]	✓	✓			79.9	91.5
Self-supervised						
ProtoNet (WRN-28-10) [30]			73.6	86.1	62.9	79.9
ProtoNet (AMDIM ResNet) [16]	✓				76.8	91.0
EPNet + SSL (WRN-28-10) [53]	✓				79.2	88.1
Semi-supervised						
LST (ResNet12) [44]	✓				70.1	78.7
PLCM (ResNet12) [37]	✓		77.6	86.1	70.1	83.7
P>M>F (IN1K, RN50)	✓		73.7	84.0	79.2	92.0
P>M>F (IN1K, ViT-Small)	✓		81.1	**92.5**	93.1	98.0
P>M>F (IN1K, ViT-base)	✓		84.3	92.2	**95.3**	**98.4**

Table 4. **miniImageNet & CIFAR** – Comparison with representative SotA FSL algorithms. Methods using external data and/or labels are indicated.

while the results for cross-domain datasets Meta-Dataset and Broader Study CDFSL are shown in Table 5 and 6 respectively. From the results we can see that our framework outperforms much state of the art in both within-domain and cross-domain conditions despite being significantly simpler than some sophisticated competitors. We remark that for the single source benchmarks in Table 4, a few competitors also used external data or ImageNet pre-training as indicated. Meanwhile our hybrid pipeline outperforms SotA pure external self-supervision [14, 27] for CDFSL in Tab 6.

4.3. Discussion

Taken together, the results show that our simple pipeline of exploiting available pre-training data and a modern architecture often outperforms sophisticated state of the art in few-shot learning. This margin is increased using our proposed adaptive fine-tuning mechanism in the meta-test stage. Based on these observations we make recommendations both for practitioners and few-shot learning researchers.

Practitioners: Increasing pre-training data size or simply using a foundation model [10, 15] and upgrading to modern architectures is likely to be more productive (and much easier to implement) than keeping up with and implementing state of the art few-shot learning algorithms. Fine-tuning is likely to be important if the target few-shot task of interest is less similar to the pre-training and meta-training data.

FSL researchers: Our results show that using external data and modern architectures is an easy and effective way to achieve strong FSL performance, and also that some SotA meta-learners fail to provide expected improvements in this regime. While external data violates definitions of the FSL problem that insist on a specific limited meta-train set, we

8 in-domain datasets	In-domain								Out-of-domain		
	INet	Omglot	Acraft	CUB	DTD	QDraw	Fungi	Flower	Sign	COCO	Avg
ProtoNet [61] (RN18)	67.01	44.5	79.56	71.14	67.01	65.18	64.88	40.26	86.85	46.48	63.29
CNAPs [52] (RN18+Adapter)	50.8	91.7	83.7	73.6	59.5	74.7	50.2	88.9	56.5	39.4	66.90
SUR [26] (RN18+Adapter)	57.2	93.2	**90.1**	82.3	73.5	81.9	67.9	88.4	67.4	51.3	75.32
T-SCNAPs [7] (RN18+Adapter)	58.8	93.9	84.1	76.8	69.0	78.6	48.8	91.6	76.1	48.7	72.64
URT [45] (RN18+Adapter)	55.7	94.4	85.8	76.3	71.8	82.5	63.5	88.2	69.4	52.2	73.98
FLUTE [60] (RN18)	51.8	93.2	87.2	79.2	68.8	79.5	58.1	91.6	58.4	50.0	71.78
URL [43] (RN18+Adapter)	57.51	94.51	88.59	80.54	76.17	81.94	68.75	92.11	63.34	54.03	75.75
ITA [42] (RN18+Adapter)	57.35	**94.96**	89.33	81.42	76.74	**82.01**	67.4	92.18	83.55	55.75	78.07
P>M>F (DINO/IN1K, RN50)	67.51	85.91	80.3	81.67	**87.08**	72.84	60.03	94.69	87.17	58.92	77.61
P>M>F (DINO/IN1K, ViT-small)	74.59	91.79	88.33	91.02	86.61	79.23	74.2	94.12	88.85	62.59	83.13
P>M>F (DINO/IN1K, ViT-base)	**77.02**	91.76	89.73	**92.94**	86.94	80.2	**78.28**	**95.79**	**89.86**	**64.97**	**84.75**
In-domain = ImageNet	In-domain	Out-of-domain									
	INet	Omglot	Acraft	CUB	DTD	QDraw	Fungi	Flower	Sign	COCO	Avg
ProtoNet [61] (RN18)	50.5	59.98	53.1	68.79	66.56	48.96	39.71	85.27	47.12	41	56.10
ALFA+FP-MAML [5] (RN12)	52.8	61.87	63.43	69.75	70.78	59.17	41.49	85.96	60.78	48.11	61.41
BOHB [54] (RN18)	51.92	67.57	54.12	70.69	68.34	50.33	41.38	87.34	51.8	48.03	59.15
CTX [24] (RN34)	62.76	82.21	79.49	80.63	75.57	72.68	51.58	**95.34**	82.65	59.9	74.28
P>M>F (DINO/IN1K, RN50)	67.08	75.33	75.39	72.08	86.42	66.79	50.53	94.14	86.54	58.2	73.25
P>M>F (DINO/IN1K, ViT-small)	74.69	80.68	76.78	85.04	86.63	71.25	54.78	94.57	88.33	62.57	77.53
P>M>F (DINO/IN1K, ViT-base)	**76.69**	**81.42**	**80.33**	**84.38**	**86.87**	**75.43**	**55.93**	95.14	**89.68**	**65.01**	**79.09**

Table 5. **Meta-Dataset** – Comparison with SotA FSL algorithms.

	ChestX			ISIC			EuroSAT			CropDisease		
	5w5s	5w20s	5w50s	5w5s	5w20s	5w50s	5w5s	5w20s	5w50s	5w5s	5w20s	5w50s
ProtoNet [55] (RN10)	24.05	28.21	29.32	39.57	49.50	51.99	73.29	82.27	80.48	79.72	88.15	90.81
RelationNet [57] (RN10)	22.96	26.63	28.45	39.41	41.77	49.32	61.31	74.43	74.91	68.99	80.45	85.08
MetaOptNet [41] (RN10)	22.53	25.53	29.35	36.28	49.42	54.80	64.44	79.19	83.62	68.41	82.89	91.76
Finetune [32] (RN10)	25.97	31.32	35.49	48.11	59.31	66.48	79.08	87.64	90.89	89.25	95.51	97.68
CHEF [1] (RN10)	24.72	29.71	31.25	41.26	54.30	60.86	74.15	83.31	86.55	86.87	94.78	96.77
STARTUP [48] (RN10)	26.94	33.19	36.91	47.22	58.63	64.16	82.29	89.26	91.99	93.02	97.51	98.45
DeepCluster2 [14, 27] (IN1K, RN50)	26.51	31.51	34.17	40.73	49.91	53.65	88.39	92.02	93.07	93.63	96.63	97.04
P>M>F (DINO/IN1K, ResNet50)	27.13	31.57	34.17	43.78	54.06	57.86	**89.18**	93.08	**96.06**	**95.06**	97.25	97.77
P>M>F (DINO/IN1K, ViT-small)	**27.27**	**35.33**	**41.39**	**50.12**	**65.78**	**73.50**	85.98	91.32	95.40	92.96	**98.12**	**99.24**

Table 6. **Broader study of cross-domain few-shot learning** – Comparison with SotA FSL algorithms.

should take this setting seriously to maintain practical relevance in the face of advancing self-supervision [15,28,38,49]. In particular, we recommend a new evaluation setting for all the standard FSL benchmarks, where pre-train data and architecture are freely chosen and clearly reported. Few-shot meta-learning methods are then evaluated on their ability to improve on linear readout, fine-tuning, or our PMF baseline for the given external dataset and architecture.

5. Conclusions

We advanced few-shot learning from the perspective of pushing the limits of a simple pre-train + ProtoNet pipeline in terms of dataset, architecture and fine-tuning strategy. We showed that source dataset, and neural architecture are dominant factors in FSL performance. When there is a domain shift between training and testing, we showed that fine-tuning the feature backbone with data augmentation is also important. We verified that our simple pipelines achieve very competitive performance in four FSL benchmarks.

Limitations and Future Work There are several limitations of our empirical study. We only scratched the surface of the impact of external data and correspondingly larger architectures on FSL. Our renewed focus on external data emphasizes the need for algorithms from the FSL community [29,41,55] to be directly compared against algorithms from the self-supervised community [10, 17], or possibly synergistically combined, as we attempt here. The hybrid pipeline that we propose is obviously restricted to modalities where large external datasets already exist, and would require significant up-front investment in compute and energy cost where pre-trained foundation models do not already exist. Possible bias within foundation models is also a potential risk [10]. Finally, while effective, our adaptive fine-tuning strategy, is rather computationally expensive at meta-test time, and may be unsupported on embedded platforms without backpropagation. Feed-forward representation adaptation methods [52] may be important for future work.

References

[1] Thomas Adler, Johannes Brandstetter, Michael Widrich, Andreas Mayr, David Kreil, Michael Kopp, Günter Klambauer, and Sepp Hochreiter. Cross-domain few-shot learning by representation fusion. In *arXiv*, 2021. 8

[2] Mohamed Afham, Salman Khan, Muhammad Haris Khan, Muzammal Naseer, and Fahad Shahbaz Khan. Rich semantics improve few-shot learning. In *BMVC*, 2021. 7

[3] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *NeurIPS*, 2020. 3

[4] Sungyong Baik, Janghoon Choi, Heewon Kim, Dohee Cho, Jaesik Min, and Kyoung Mu Lee. Meta-learning with task-adaptive loss function for few-shot learning. In *ICCV*, 2021. 1

[5] Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. In *NeurIPS*, 2020. 8

[6] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022. 4

[7] Peyman Bateni, Jarred Barber, Jan-Willem van de Meent, and Frank Wood. Enhancing few-shot image classification with unlabelled examples. In *WACV*, 2022. 7, 8

[8] Luca Bertinetto, João F. Henriques, Philip H.S. Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *ICLR*, 2019. 4, 5

[9] Luca Bertinetto, Joao F. Henriques, Jack Valmadre, Philip H. S. Torr, and Andrea Vedaldi. Learning feed-forward one-shot learners. In *NIPS*, 2016. 1, 2

[10] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2, 3, 6, 7, 8

[11] Myriam Bontonou, Nicolas Farrugia, and Vincent Gripon. Few-shot learning for decoding brain signals. *CoRR*, abs/2010.12500, 2020. 2

[12] Stevo Bozinovski. Reminder of the first paper on transfer learning in neural networks, 1976. *Informatica*, 44(3), 2020. 3

[13] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020. 3

[14] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments, 2020. 7, 8

[15] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. 3, 4, 5, 6, 7, 8

[16] Da Chen, Yuefeng Chen, Yuhong Li, Feng Mao, Yuan He, and Hui Xue. Self-supervised learning for few-shot image classification. In *ICASSP*, 2021. 7

[17] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 3, 5, 6, 8

[18] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, 2020. 1, 3, 5, 6

[19] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *ICLR*, 2019. 1, 2, 7

[20] Yinbo Chen, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *ICCV*, 2021. 1, 3, 7

[21] Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 3

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *ACL*, 2019. 3, 4

[23] Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 1, 2, 3, 7

[24] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. In *NeurIPS*, 2021. 3, 5, 8

[25] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2, 4

[26] Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Selecting relevant features from a multi-domain representation for few-shot classification. In *ECCV*, 2020. 8

[27] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In *CVPR*, 2021. 3, 5, 7, 8

[28] Linus Ericsson, Henry Gouk, Chen Change Loy, and Timothy M Hospedales. Self-supervised representation learning: Introduction, advances and challenges. *IEEE Signal Processing Magazine*, 2022. 3, 8

[29] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 2, 3, 6, 8

[30] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *ICCV*, 2019. 3, 7

[31] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *ICCV*, 2019. 2, 3

[32] Yunhui Guo, Noel C Codella, Leonid Karlinsky, James V Codella, John R Smith, Kate Saenko, Tajana Rosing, and Rogerio Feris. A broader study of cross-domain few-shot learning. In *ECCV*, 2020. 3, 4, 5, 8

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[34] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[35] Shell Xu Hu, Pablo Moreno, Yang Xiao, Xi Shen, Guillaume Obozinski, Neil Lawrence, and Andreas Damianou. Empirical bayes transductive meta-learning with synthetic gradients. In *ICLR*, 2020. 7

[36] Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Leveraging the feature distribution in transfer-based few-shot learning. In *ICANN*, 2021. 7

[37] Kai Huang, Jie Geng, Wen Jiang, Xinyang Deng, and Zhe Xu. Pseudo-loss confidence metric for semi-supervised few-shot learning. In *ICCV*, 2021. 5, 7

[38] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 2021. 3, 8

[39] Arman Kazemi, Shubham Sahay, Ayush Saxena, Mohammad Mehdi Sharifi, Michael Niemier, and X. Sharon Hu. A flash-based multi-bit content-addressable memory with euclidean squared distance. In *IEEE/ACM International Symposium on Low Power Electronics and Design*, 2021. 2

[40] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 2015. 1

[41] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 1, 2, 6, 7, 8

[42] Wei-Hong Li, Xialei Liu, and Hakan Bilen. Improving task adaptation for cross-domain few-shot learning. *arXiv preprint arXiv:2107.00358*, 2021. 4, 7, 8

[43] Wei-Hong Li, Xialei Liu, and Hakan Bilen. Universal representation learning from multiple domains for few-shot classification. In *ICCV*, 2021. 8

[44] Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. Learning to self-train for semi-supervised few-shot classification. *NeurIPS*, 2019. 5, 7

[45] Lu Liu, William Hamilton, Guodong Long, Jing Jiang, and Hugo Larochelle. A universal representation transformer layer for few-shot image classification. In *ICLR*, 2021. 8

[46] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *ICLR*, 2019. 5

[47] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*, 2020. 1, 2

[48] Cheng Perng Phoo and Bharath Hariharan. Self-training for few-shot transfer across extreme task differences. In *ICLR*, 2021. 8

[49] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 4, 8

[50] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1, 2

[51] Mengye Ren, Eleni Triantafillou, Jake Snell Sachin Ravi, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 1, 5

[52] James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E. Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *NeurIPS*, 2020. 8

[53] Pau Rodríguez, Issam Laradji, Alexandre Drouin, and Alexandre Lacoste. Embedding propagation: Smoother manifold for few-shot classification. In *ECCV*, 2020. 7

[54] Tonmoy Saikia, Thomas Brox, and Cordelia Schmid. Optimized generic feature learning for few-shot classification across domains. In *arXiv*, 2020. 8

[55] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017. 1, 2, 3, 4, 6, 7, 8

[56] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017. 2, 3

[57] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 1, 8

[58] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Commun. ACM*, 59(2):64–73, jan 2016. 2

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

[60] Eleni Triantafillou, Hugo Larochelle, Richard Zemel, and Vincent Dumoulin. Learning a universal template for few-shot dataset generalization. In *ICML*, 2021. 8

[61] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *ICLR*, 2020. 2, 3, 4, 5, 7, 8

[62] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *NeurIPS*, 2016. 1, 3, 4, 5

[63] Yan Wang, Wei-Lun Chao, Kilian Q. Weinberger, and Laurens van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning, 2019. 1, 2

[64] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020. 1, 2

[65] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014. 2, 3

[66] X. Zhang, D. Meng, H. Gouk, and T. Hospedales. Shallow bayesian meta learning for real-world few-shot recognition. In *ICCV*, 2021. 1, 2, 6