International Journal of Computer Vision (2021) 129:1930–1953
https://doi.org/10.1007/s11263-020-01381-4

Learning Adaptive Classifiers Synthesis for Generalized Few-Shot Learning

Han-Jia Ye[1] · **Hexiang Hu**[2] · **De-Chuan Zhan**[1]

Received: 21 December 2019 / Accepted: 3 September 2020 / Published online: 19 April 2021

Abstract
Object recognition in the real-world requires handling long-tailed or even open-ended data. An ideal visual system needs to recognize the populated head visual concepts reliably and meanwhile efficiently learn about emerging new tail categories with a few training instances. Class-balanced many-shot learning and few-shot learning tackle one side of this problem, by either learning strong classifiers for head or learning to learn few-shot classifiers for the tail. In this paper, we investigate the problem of *generalized few-shot learning (GFSL)*—a model during the deployment is required to learn about tail categories with few shots and simultaneously classify the head classes. We propose the ClAssifier SynThesis LEarning (CASTLE), a learning framework that learns how to synthesize calibrated few-shot classifiers in addition to the multi-class classifiers of head classes with a shared neural dictionary, shedding light upon the *inductive* GFSL. Furthermore, we propose an adaptive version of CASTLE (ACASTLE) that adapts the head classifiers conditioned on the incoming tail training examples, yielding a framework that allows effective backward knowledge transfer. As a consequence, ACASTLE can handle GFSL with classes from heterogeneous domains effectively. CASTLE and ACASTLE demonstrate superior performances than existing GFSL algorithms and strong baselines on *Mini*ImageNet as well as *Tiered*ImageNet datasets. More interestingly, they outperform previous state-of-the-art methods when evaluated with standard few-shot learning criteria.

Keywords Image recognition · Meta learning · Generalized few-shot learning · Few-shot learning · Recognition with heterogeneous visual domain

1 Introduction

Visual recognition for objects in the "long tail" has been an important challenge to address (Wang et al. 2017; Liu et al. 2019; Kang et al. 2020; Zhou et al. 2020). We often have a very limited amount of data on those objects as they are infrequently observed and/or visual exemplars of them are hard to collect. As such, state-of-the-art methods (e.g., deep learning) can not be directly applied due to their notorious demand of a large number of annotated data (Krizhevsky et al. 2017; Simonyan and Zisserman 2014; He et al. 2016).

Few-shot learning (FSL) (Vinyals et al. 2016; Larochelle 2018) is mindful of the limited data per tail concept (i.e., shots), which attempts to address this challenging problem by distinguishing between the data-rich head categories as SEEN classes and data-scarce tail categories as UNSEEN classes. While it is difficult to build classifiers with data from UNSEEN classes, FSL mimics the test scenarios by sampling few-shot tasks from SEEN class data, and extracts inductive biases for effective classifiers acquisition on UNSEEN ones. Instance embedding (Vinyals et al. 2016; Snell et al. 2017; Rusu et al. 2019; Ye et al. 2020), model initialization (Finn et al. 2017; Nichol et al. 2018; Antoniou et al. 2019), image generator (Wang et al. 2018), and optimization flow (Ravi and Larochelle 2017; Lee et al. 2019) act as popular meta-knowledge and usually incorporates with FSL.

This type of learning makes the classifier from few-shot learning for UNSEEN classes difficult to be combined directly

Han-Jia Ye and Hexiang Hu have contributed equally to this study.

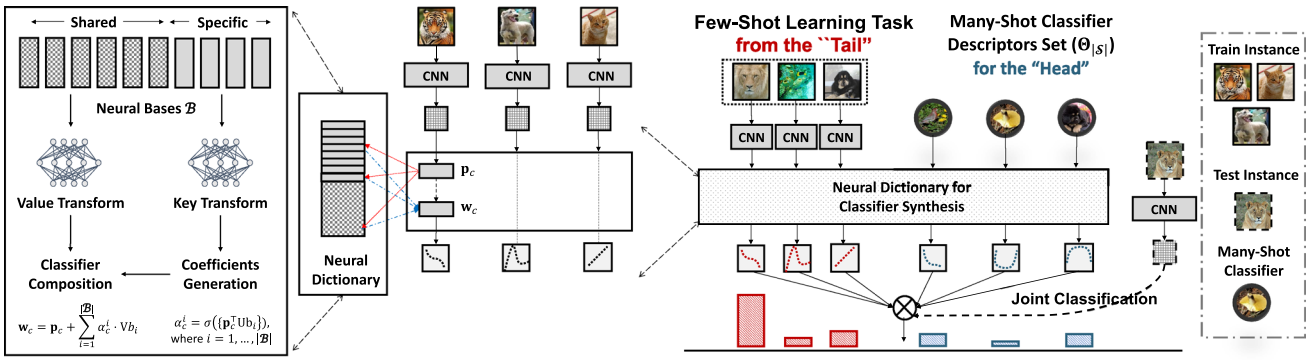
✉ Han-Jia Ye
yehj@lamda.nju.edu.cn

Hexiang Hu
hexiangh@usc.edu

De-Chuan Zhan
zhandc@lamda.nju.edu.cn

[1] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China

[2] University of Southern California, Los Angeles, USA



(a) **Classifier Synthesis** using Few-Shot Training Data (b) **Inference** with Generalized Few-Shot Learning

Fig. 1 A conceptual diagram comparing the Few-Shot Learning (FSL) and the Generalized Few-Shot Learning (GFSL).GFSL requires to extract inductive bias from SEEN categories to facilitate efficiently learning on few-shot UNSEEN tail categories, while maintaining discernability on head classes

with the classifier from many-shot learning for SEEN classes, however, the demand to recognize *all* object categories simultaneously in object recognition is essential as well.

In this paper, we study the problem of *Generalized Few-Shot Learning* (GFSL), which focuses on the *joint* classification of both data-rich and data-poor categories. Figure 1 illustrates the high-level idea of the GFSL, contrasting the standard FSL. In particular, our goal is for the model trained on the SEEN categories to be capable of incorporating the limited UNSEEN class examples, and make predictions for test data in both the head and tail of the entire distribution of categories.

One naive GFSL solution is to train a single classifier over the imbalanced long-tail distribution (Hariharan and Girshick 2017; Wang et al. 2017; Liu et al. 2019; Zhou et al. 2020), and re-balance it (Cui et al. 2019; Cao et al. 2019; Kang et al. 2020). One main advantage of such a joint learning objective over all classes is that it characterizes both SEEN and UNSEEN classes simultaneously. In other words, training of one part (e.g., head) naturally takes the other part (e.g., tail) into consideration, and promotes the knowledge transfer between classes. However, such a transductive learning paradigm requires collecting the limited tail data in advance, which is violated in many real-world tasks. In contrast to it, our learning setup requires an *inductive* modeling of the tail, which is therefore more challenging as we assume no knowledge about the UNSEEN tail categories is available during the model learning phase.

There are two main challenges in the inductive GFSL problem, including how to construct the many-shot and few-shot classifiers in the GFSL scenario and how to *calibrate* their predictions.

First, the head and tail classifiers for a GFSL model should encode different properties of all classes towards high discerning ability, and the classifiers for the many-shot part should be adapted based on the tail concepts accordingly. For example, if the UNSEEN classes come from different domains, the same single SEEN classifier is difficult to handle their diverse properties and should not be left alone in this dynamic process. Furthermore, as observed in the generalized zero-shot learning scenario (Chao et al. 2016), a classifier performs over-confident with its familiar concepts and fear to make predictions for those UNSEEN ones, which leads to a confidence gap when predicting SEEN and UNSEEN classes. The calibration issue appears in the generalized few-shot learning as well, i.e., SEEN and UNSEEN classifiers have different confidence ranges. We empirically find that directly optimizing two objectives together could not resolve the problem completely.

To this end, we propose *ClAssifier SynThesis LEarning* (CASTLE), where the few-shot classifiers are synthesized based on a neural dictionary with common characteristics across classes. Such synthesized few-shot classifiers *are then used together* with the many-shot classifiers, and learned end-to-end. To this purpose, we create a learning scenario by sampling a set of data instances from SEEN categories and pretend that they come from UNSEEN categories, and apply the synthesized classifiers (based on the above instances) as if they are many-shot classifiers to optimize multi-class classification together with the remaining many-shot SEEN classifiers. In other words, we construct few-shot classifiers to *not only perform well on the few-shot classes but also to be competitive when used in conjunction with many-shot classifiers of populated classes*. We argue that such highly contrastive learning can benefit the few-shot classification in two aspects: (1) it provides high discernibility for its synthesized classifiers. (2) it makes the synthesized classifier automatically calibrated with the many-shot classifiers.

Taking steps further, we then propose the *Adaptive ClAssifier SynThesis LEarning* (ACASTLE), with additional flexibility to adapt the many-shot classifiers based on few-shot training examples. As a result, it allows backward knowledge

transfer (Lopez-Paz and Ranzato 2017)—new knowledge learned from novel few-shot training examples can benefit the existing many-shot classifiers. In ACASTLE, the neural dictionary is the concatenation of the shared and the task-specific neural bases, whose elements summarize the generality of all visual classes and the specialty of current few-shot categories. This improved neural dictionary facilitates the adaptation of the many-shot classifiers conditioned on the limited tail training examples. The adapted many-shot classifiers in ACASTLE *are then used together* with the (jointly) synthesized few-shot classifiers for GFSL classification.

We first verify the effectiveness of the synthesized GFSL classifiers over multi-domain GFSL tasks, where the UNSEEN classes would come from diverse domains. ACASTLE can best handle such task heterogeneity due to its ability to adapt the head classifiers. Next, we empirically validate our approach on two standard benchmark datasets—*Mini*ImageNet (Vinyals et al. 2016) and *Tiered*ImageNet (Ren et al. 2018). The proposed approach retains competitive tail concept recognition performances while outperforming existing approaches on *generalized* few-shot learning with criteria from different aspects. By carefully selecting a prediction bias from the validation set, those miscalibrated FSL approaches or other baselines perform well in the GFSL scenario. The implicit confidence calibration in CASTLE and ACASTLE works as well as or even better than the post-calibration techniques. We note that CASTLE and ACASTLE are applicable for standard few-shot learning, which stays competitive with and sometimes even outperforms state-of-the-art methods when evaluated on two popular FSL benchmarks.

Our contributions are summarized as follows:

- We propose a framework that synthesizes few-shot classifiers for GFSL with a shared neural dictionary, as well as its adaptive variant that modifies SEEN many-shot classifiers to allow the backward knowledge transfer.
- We extend an existing GFSL learning framework into an end-to-end counterpart that learns and contrasts the few-shot and the many-shot classifiers simultaneously, which is observed beneficial to the confidence calibration of these two types of classifiers.
- We empirically demonstrate that ACASTLE is effective in backward transferring knowledge when learning novel classes under the setting of multi-domain GFSL. Meanwhile, we perform a comprehensive evaluation of both existing and our approaches with criteria from various perspectives on multiple GFSL benchmarks.

In the rest sections of this paper, we first describe the problem formulation of GFSL in Sect. 2, and then introduce our CASTLE/ACASTLE approach in Sect. 3. We conduct thorough experiments (see Sect. 4 for the setups) to verify the the proposed CASTLE and ACASTLE across multiple benchmarks. We first conduct a pivot study on multi-domain GFSL benchmarks (Sect. 5) to study the backward transfer capability of different methods. Then we evaluate both ACASTLE and CASTLE on popular GFSL (Sect. 6.3), and FSL benchmarks (Sect. 6.4). Eventually, we review existing related works in Sect. 7 and discuss the connections to our work.

2 Problem Description

We define a K-shot N-way classification task to be one with N classes to make prediction and K training examples per class for learning. The training set (i.e., the support set) is represented as $\mathcal{D}_{\mathbf{train}} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{NK}$, where $\mathbf{x}_i \in \mathbb{R}^D$ is an instance and $\mathbf{y}_i \in \{0, 1\}^N$ (i.e., one-hot vector) is its label. Similarly, the test set (*a.k.a.* the query set) is $\mathcal{D}_{\mathbf{test}}$, which contains *i.i.d.* samples from the same distribution as $\mathcal{D}_{\mathbf{train}}$.

2.1 Meta-Learning for Few-Shot Learning (FSL)

In *many-shot learning*, where K is large, a classification model $f : \mathbb{R}^D \rightarrow \{0, 1\}^N$ is learned by optimizing over the instances from the head classes:

$$\mathbb{E}_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_{\mathbf{train}}} \ell(f(\mathbf{x}_i), \mathbf{y}_i)$$

Here f is often instantiated as an embedding function $\phi(\cdot) : \mathbb{R}^D \rightarrow \mathbb{R}^d$ and a linear classifier $\mathbf{\Theta} \in \mathbb{R}^{d \times N}$: $f(\mathbf{x}_i) = \phi(\mathbf{x}_i)^\top \mathbf{\Theta}$. We do not consider the bias term in the linear classifier in the following discussions, and the weight vector of the class n is denoted as $\mathbf{\Theta}_n$. The loss function $\ell(\cdot, \cdot)$ measures the discrepancy between the prediction and the true label.

On the other hand, *Few-shot learning (FSL)* faces the challenge in transferring knowledge across learning visual concepts from head to the tail. It assumes two non-overlapping sets of SEEN ($\mathcal{S}$) and UNSEEN ($\mathcal{U}$) classes. During training, it has access to all SEEN classes for learning an inductive bias, which is then transferred to learn a good classifier on $\mathcal{U}$ rapidly with a small K.

In summary, we aim to minimize the following expected error in FSL:

$$\mathbb{E}_{\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}} \mathbb{E}_{(\mathbf{x}_j, \mathbf{y}_j) \in \mathcal{D}^{\mathcal{U}}_{\mathbf{test}}} \left[\ell \left(f \left(\mathbf{x}_j; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}} \right), \mathbf{y}_j \right) \right] \tag{1}$$

Given any UNSEEN few-shot training set $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$, the function f in Eq. 1 maps $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ to the classifiers of UNSEEN classes, which achieves low error of classifying instances in $\mathcal{D}^{\mathcal{U}}_{\mathbf{test}}$ via the inference $f\left(\mathbf{x}_j; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}}\right)$. Here instances in $\mathcal{D}^{\mathcal{U}}_{\mathbf{test}}$ are sampled from the same set of classes as $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$.

Since we do not have access to the UNSEEN classes during the model training, meta-learning becomes an effective framework for FSL (Vinyals et al. 2016; Finn et al. 2017; Snell et al. 2017) in the recent years. In particular, a K-shot N-way task $\mathcal{D}^{\mathcal{S}}_{\mathbf{train}}$ sampled from $\mathcal{S}$ is constructed by randomly choosing N categories from $\mathcal{S}$ and K examples in each of them. [1] The main idea of meta-learning is to *mimic* the future few-shot learning scenario by optimizing a shared f across K-shot N-way sampled tasks drawn from the SEEN class sets $\mathcal{S}$.

$$\min_{f} \mathbb{E}_{(\mathcal{D}^{\mathcal{S}}_{\mathbf{train}}, \mathcal{D}^{\mathcal{S}}_{\mathbf{test}})\sim\mathcal{S}} \mathbb{E}_{(\mathbf{x}_j,\mathbf{y}_j)\in\mathcal{D}^{\mathcal{S}}_{\mathbf{test}}} \left[\ell \left(f\left(\mathbf{x}_j; \mathcal{D}^{\mathcal{S}}_{\mathbf{train}}\right), \mathbf{y}_j \right) \right] \tag{2}$$

Equation 2 approximates the Eq. 1 with the SEEN class data, and the meta-model f is applied to different few-shot tasks constructed by the data of SEEN classes. Following this split use of $\mathcal{S}$, tasks and classes related to $\mathcal{S}$ are denoted as "meta-training", and called "meta-val/test" when they are related to $\mathcal{U}$. Similar to Eq. 1, a corresponding test set $\mathcal{D}^{\mathcal{S}}_{\mathbf{test}}$ is sampled from the N classes in $\mathcal{S}$ to evaluate the resulting few-shot classifier $f\left(\cdot\,; \mathcal{D}^{\mathcal{S}}_{\mathbf{train}}\right)$. Therefore, we expect the learned classifier "generalizes" well on the training few-shot tasks sampled from SEEN classes, to "generalize" well on few-shot tasks drawn from UNSEEN class set $\mathcal{U}$. Once we learned f, for a few-shot task $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ with unseen classes $\mathcal{U}$, we can get its classifier $f\left(\cdot\,; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}}\right)$ as Eq. 2.

Specifically, one popular form of the meta-knowledge to transfer between SEEN and UNSEEN classes is the instance embedding, i.e., $f = \phi$, which transforms input examples into a latent space with d dimensions (Vinyals et al. 2016; Snell et al. 2017). ϕ is learned to pull similar objects close while pushing dissimilar ones far away (Koch et al. 2015). For a test instance $\mathbf{x}_j$, the embedding function ϕ makes a prediction based on a soft nearest neighbor classifier:

$$\begin{aligned}\hat{y}_j &= f\left(\mathbf{x}_j; \mathcal{D}_{\mathbf{train}}\right) \\ &= \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}_{\mathbf{train}}} \mathbf{sim}\left(\phi(\mathbf{x}_j), \phi(\mathbf{x}_i)\right) \cdot \mathbf{y}_i\end{aligned}$$

$\mathbf{sim}(\phi(\mathbf{x}_j), \phi(\mathbf{x}_i))$ measures the similarity between the test instance $\phi(\mathbf{x}_j)$ and each training instance $\phi(\mathbf{x}_i)$. When there is more than one instance per class, i.e., $K > 1$, instances in the same class can be averaged to assist make a final decision (Snell et al. 2017). By learning a good ϕ, important visual features for few-shot classification are distilled, which helps the few-shot tasks with classes from the UNSEEN classes.

[1] We use the super-script $\mathcal{S}$ and $\mathcal{U}$ to denote a set or an instance sampled from $\mathcal{S}$ and $\mathcal{U}$, respectively.

2.2 Meta-learning for Generalized Few-Shot Learning (GFSL)

Different from FSL which neglects classification of the $\mathcal{S}$ classes, *Generalized Few-Shot Learning (GFSL)* aims at building a model that simultaneously predicts over $\mathcal{S} \cup \mathcal{U}$ categories. As a result, such a model needs to deal with many-shot classification from $|\mathcal{S}|$ SEEN classes along side with learning $|\mathcal{U}|$ emerging UNSEEN classes. [2] In *inductive* GFSL, the model only has access to the head part $\mathcal{S}$ and is required to extract knowledge which facilitates building a joint classifier over SEEN and UNSEEN categories once with limited tail examples.

In GFSL, we require the function f to map from a few-shot training set $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ to a classifier classifying both SEEN and UNSEEN classes, which means a GFSL classifier f should have a low expected error as what follows:

$$\mathbb{E}_{\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}} \mathbb{E}_{(\mathbf{x}_j,\mathbf{y}_j)\in\mathcal{D}^{\mathcal{S}\cup\mathcal{U}}_{\mathbf{test}}} \left[\ell \left(f\left(\mathbf{x}_j; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}}, \mathbf{\Theta}_{\mathcal{S}}\right), \mathbf{y}_j \right) \right] \tag{3}$$

Different from Eq. 1, in the GFSL setting, the meta-model f generates classifier $f\left(\cdot; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}}, \mathbf{\Theta}_{\mathcal{S}}\right)$ through taking both the UNSEEN class few-shot training set $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ and a class descriptors set $\mathbf{\Theta}_{\mathcal{S}}$ summarizing the information of the SEEN classes as input. Besides, such classifier is able to tell instances from the joint set of $\mathcal{S} \cup \mathcal{U}$.

Similarly, we simulate many GFSL tasks from the SEEN classes. At each time, we split the SEEN classes into a tail split with classes $\mathcal{C}$, and treat remaining $|\mathcal{S}| - |\mathcal{C}|$ classes as the head split. Eq. 3 is transformed into:

$$\min_{f} \sum_{\mathcal{C}\subset\mathcal{S}} \sum_{(\mathbf{x}_j,\mathbf{y}_j)\sim\mathcal{S}} \ell \left(f\left(\mathbf{x}_j; \mathcal{D}^{\mathcal{C}}_{\mathbf{train}}, \mathbf{2}_{\mathcal{S}-\mathcal{C}}\right), \mathbf{y}_j \right) \tag{4}$$

In particular, the function f outputs a $\mathcal{S}$-way classifier with two steps: (1) For the tail split $\mathcal{C}$, it follows what f does in Section 2 and generates the classifiers of $\mathcal{C}$ using their few-shot training examples $\mathcal{D}^{\mathcal{C}}_{\mathbf{train}}$. (2) For the head split $\mathcal{S} - \mathcal{C}$, this function directly make use of the many-shot classifiers of the $\mathcal{S} - \mathcal{C}$ classes to generate the classifiers (instead of asking for training examples of head split).

3 Method

There are two key components in CASTLE and ACASTLE. First, it presents an effective learning algorithm that learns many-shot classifiers and few-shot classifiers simultaneously, in an end-to-end manner. Second, it contains a classifier composition model, which synthesizes classifiers for the

[2] $|\mathcal{S}|$ and $|\mathcal{U}|$ denote the total number of classes from the SEEN and UNSEEN class sets, respectively.

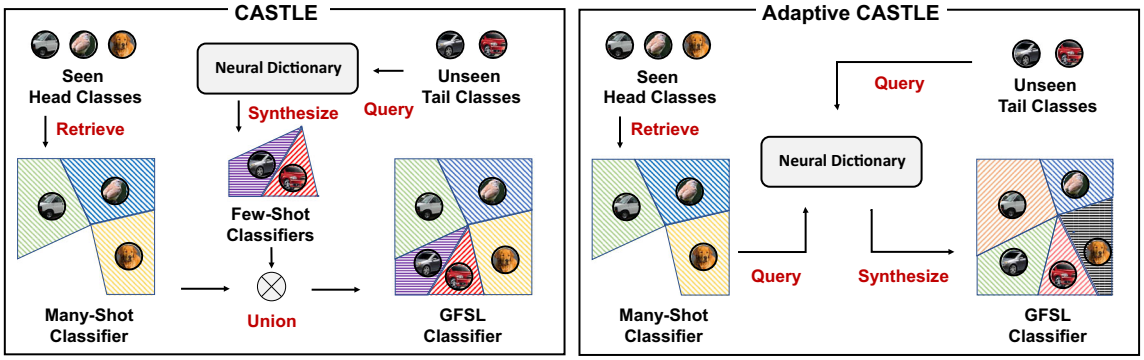


Fig. 2 Illustration of adaptive GFSL learning process of CASTLE and ACASTLE. Different from the stationary learning process (l.h.s.) of CASTLE, ACASTLE (r.h.s.) synthesizes the GFSL classifiers for SEEN and UNSEEN classes in an adaptive manner—the many-shot classifiers of head classes are also conditioned on the training instances from the tail classes

tail classes using the few-shot training data, via querying a learnable neural dictionary.

In Sect. 3.1, we utilize the objective in Eq. 3 that directly contrasts many-shot classifiers with few-shot classifiers, via constructing classification tasks over $\mathcal{U} \cup \mathcal{S}$ categories. By reusing the parameters of the many-shot classifier, the learned model calibrates the prediction ranges over head and tail classes naturally. It enforces the few-shot classifiers to explicitly compete against the many-shot classifiers in the model learning, which leads to more discriminative few-shot classifiers in the GFSL setting. In Sect. 3.2, we introduce the classifier composition model uses a few-shot training data to query the neural bases, and then assemble the target "synthesized classifiers". CASTLE sets a shared neural bases across tasks, which keeps stationary many-shot classifiers all the time; while with both shared and specific components in the neural dictionary, the SEEN class classifiers will be adapted based on its relationship with UNSEEN class instances in ACASTLE.

3.1 Unified Learning of Few-Shot and Many-Shot Classifiers

In addition to transferring knowledge from SEEN to UNSEEN classes as in FSL, in generalized few-shot learning, the few-shot classifiers is required to do well when used in conjunction with many-shot classifiers. Suppose we have sampled a K-shot N-way few-shot learning task $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$, which contains $|\mathcal{U}|$ visual UNSEEN categories, a GFSL classifier f should have a low expected error as in Eq. 3

The set of "class descriptors" $\boldsymbol{\Theta}$ of a classifier is a set of vectors summarizes the characteristic of its target classes, e.g., some preserved instances from those classes. For the SEEN classes $\mathcal{S}$, we set the descriptors as the union of the weight vectors in the many-shot classifiers $\boldsymbol{\Theta}_{\mathcal{S}} = \{\boldsymbol{\Theta}_s\}_{s\in\mathcal{S}}$ (i.e., the liner classifier over the embedding function $\phi(\cdot)$). For each task, the classifier f predicts a test instance in $\mathcal{D}^{\mathcal{S}\cup\mathcal{U}}_{\mathbf{test}}$ towards *both* tail classes $\mathcal{U}$ and head classes $\mathcal{S}$. In other words, based on $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ and the class descriptors set of the many-shot classifier $\boldsymbol{\Theta}_{\mathcal{S}}$, a randomly sampled instance in $\mathcal{S}\cup\mathcal{U}$ should be effectively predicted. In summary, a GFSL classifier generalizes its joint prediction ability to $\mathcal{S}\cup\mathcal{U}$ given $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ and $\boldsymbol{\Theta}_{\mathcal{S}}$ during inference.

3.1.1 Neural Dictionary for Classifier Synthesis

We use neural dictionary to implement the joint prediction $f\left(\mathbf{x}_j; \mathcal{D}^{\mathcal{U}}_{\mathbf{train}}, \boldsymbol{\Theta}_{\mathcal{S}}\right)$ in Eq. 3. A neural dictionary is a module with a set of neural bases $\mathcal{B}$, which represents its input as a weighted combination of those bases based on their similarities. To classify an instance during inference, the neural dictionary takes partial or both of the limited tail instances $\mathcal{D}^{\mathcal{U}}_{\mathbf{train}}$ and the context of the SEEN classifiers descriptors set $\boldsymbol{\Theta}_{\mathcal{S}}$ into account, and synthesizes the classifier for the corresponding classes with $\mathcal{B}$. The details of the neural dictionary will be described in the next subsection.

3.1.2 Unified Learning Objective

ACASTLE and its variants learn a generalizable GFSL classifier via training on the SEEN class set $\mathcal{S}$. We sample a "fake" K-shot N-way few-shot task from $\mathcal{S}$, which contains categories $\mathcal{C}$. Given the "fake" few-shot task, we treat the remaining $\mathcal{S}-\mathcal{C}$ classes as the "fake" head classes, whose corresponding many-shot classifier descriptors set is $\boldsymbol{\Theta}_{\mathcal{S}-\mathcal{C}}$. Then the GFSL model needs to build a classifier targets any instance in $\mathcal{C}\cup(\mathcal{S}-\mathcal{C})$. As mentioned before, we *synthesize* both the few-shot classifiers for $\mathcal{C}$ by $\mathbf{W}_{\mathcal{C}} = \{\ \mathbf{w}_c \mid c\in\mathcal{C}\ \}$ and the many-shot classifier $\hat{\boldsymbol{\Theta}}_{\mathcal{S}-\mathcal{C}} = \{\ \hat{\boldsymbol{\Theta}}_c \mid c\in\mathcal{S}-\mathcal{C}\ \}$ with a neural dictionary, so that the composition of one classifier will consider the context of others.

Both the synthesized many-shot classifier (from the "fake" many-shot classes $\mathcal{S}-\mathcal{C}$) $\hat{\mathbf{\Theta}}_{\mathcal{S}-\mathcal{C}}$ and few-shot classifier (from the "fake" few-shot classes $\mathcal{C}$) $\mathbf{W}_{\mathcal{C}}$ are combined together to form a joint classifier $\hat{\mathbf{W}} = \mathbf{W}_{\mathcal{C}} \cup \hat{\mathbf{\Theta}}_{\mathcal{S}-\mathcal{C}}$, over *all* classes in $\mathcal{S}$.

Finally, we optimize the learning objective as follows:

$$\min_{\{\phi,\mathcal{B},\{\mathbf{\Theta}_s\},\mathrm{U},\mathrm{V}\}} \sum_{\mathcal{C}\subset\mathcal{S}} \sum_{(\mathbf{x}_j,\mathbf{y}_j)\sim\mathcal{S}} \ell\Big(\hat{\mathbf{W}}^{\top}\phi(\mathbf{x}_j), \mathbf{y}_j\Big) \tag{5}$$

In addition to the learnable neural bases $\mathcal{B}$, U and V are two projections in the neural bases to facilitate the synthesis of the classifier, and there is no bias term in our implementation. Despite that the few-shot classifiers $\mathbf{W}_{\mathcal{C}}$ are synthesized using with K training instances, they are optimized to perform well on all the instances from $\mathcal{C}$ and moreover, to perform well against all the instances from other SEEN categories. The many-shot classifiers $\mathbf{\Theta}_{\mathcal{S}}$ are not stationary, which are *adapted* based on the context of the current few-shot instances (the adaptive GFSL classifier notion is illustrated in Fig. 2). Note that $\mathbf{W}_{\mathcal{C}}$ and $\hat{\mathbf{\Theta}}_{\mathcal{S}-\mathcal{C}}$ are synthesized based on the neural dictionary, which serves as the **bridge** to connect the "fake" few-shot class set $\mathcal{C}$ and the "fake" many-shot class set $(\mathcal{S}-\mathcal{C})$.

After minimizing the accumulated loss in Eq. 5 over multiple GFSL tasks, the learned model extends its discerning ability to UNSEEN classes so that has low error in Eq. 3. During inference, ACASTLE synthesizes the classifiers for UNSEEN classes based on the neural dictionary with their few-shot training examples, and makes a joint prediction over $\mathcal{S}\cup\mathcal{U}$ with the help of the *adapted* many-shot classifier $\hat{\mathbf{\Theta}}_{\mathcal{S}}$.

3.1.3 Reuse Many-Shot Classifiers

We optimize Eq. 5 by using the many-shot classifier over $\mathcal{S}$ to initialize the embedding ϕ. In detail, a $|\mathcal{S}|$-way many-shot classifier is trained over all SEEN classes with the cross-entropy loss, whose backbone is used to initialize the embedding ϕ in the GFSL classifier. We empirically observed that such initialization is essential for the prediction calibration between SEEN and UNSEEN classes, more details could be found in "Appendix 1" and "Appendix 4".

3.1.4 Multi-classifier Learning

A natural way to minimize Eq. 5 implements a stochastic gradient descent step in each mini-batch by sampling one GFSL task, which contains a K-shot N-way training set together with a set of test instances $(\mathbf{x}_j, \mathbf{y}_j)$ from $\mathcal{S}$. It is clear that increasing the number of GFSL tasks per gradient step can improve the optimization stability. Therefore, we propose an efficient implementation that utilizes *a large number of* GFSL tasks to compute gradients. Specifically, we sample two sets of instances from *all* SEEN classes, i.e., $\mathcal{D}^{\mathcal{S}}_{\mathbf{train}}$ and $\mathcal{D}^{\mathcal{S}}_{\mathbf{test}}$. Then we construct a large number of joint classifiers $\{\hat{\mathbf{W}}^z = \mathbf{W}^z_{\mathcal{C}} \cup \hat{\mathbf{\Theta}}^z_{\mathcal{S}-\mathcal{C}} \mid z = 1,\ldots,Z\}$ with different sets of $\mathcal{C}$, which is then applied to compute the averaged loss over z using Eq. 5. Note that there is only one single forward step to get the embeddings of the involved instances, and we mimic multiple GFSL tasks through different random partitions of the "fake" few-shot and "fake" many-shot classes. In the scope of this paper, ACASTLE variants always use *multi-classifier learning* unless it is explicitly mentioned. With this, we observed a significant speed-up in terms of convergence (see "Appendix 10" for the ablation study).

3.2 Classifier Composition with a Neural Dictionary

Neural dictionary is an essential module for classifier composition in ACASTLE variants. We describe the composition of the neural dictionary and the way to synthesize tail classifiers first, followed by the adaption of the head classifiers. The neural dictionary formalizes both head and tail classifiers with common bases, which benefits the relationship transition between classes. Furthermore, the neural dictionary encodes the shared primitives for composing classifiers, which serves as a kind of meta-knowledge to be transferred across both the SEEN and the UNSEEN classes.

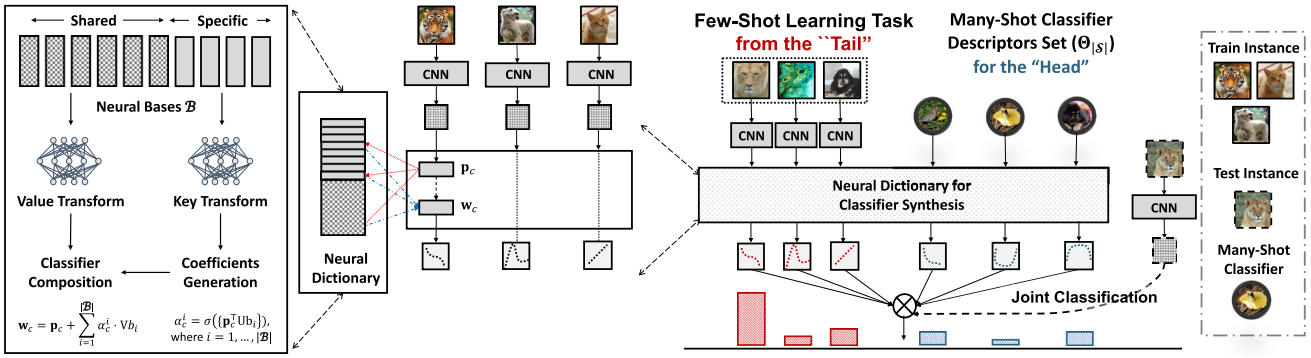
Similar to Vaswani et al. (2017), we define a neural dictionary as pairs of learnable "key" and "value" embeddings, where each "key" and "value" is associated with a set of neural bases, which are designed to encode shared primitives for composing the classifier of $\mathcal{S}\cup\mathcal{U}$. Formally, the neural bases contain two sets of elements:

$$\mathcal{B} = \mathcal{B}_{\mathbf{share}} \bigcup \mathcal{B}_{\mathbf{specific}}$$

$\mathcal{B}_{\mathbf{share}}$ contains a set of $|\mathcal{B}_{\mathbf{share}}|$ learnable bases $\mathcal{B}_{\mathbf{share}} = \{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{|\mathcal{B}_{\mathbf{share}}|}\}$, and $\mathbf{b}_k \in \mathcal{B}_{\mathbf{share}} \in \mathbb{R}^d$. This part in the neural dictionary is shared when synthesizing classifiers for different kinds of tasks. $\mathcal{B}_{\mathbf{specific}}$ characterizes the local information of the input to the neural dictionary, i.e., the training set $\mathcal{D}_{\mathbf{train}}$ of the current few-shot task with tail classes and the descriptors set of the many-shot classifier.

The key and value for the neural dictionary are generated based on two linear projections $\mathrm{U} \in \mathbb{R}^{d\times d}$ and $\mathrm{V} \in \mathbb{R}^{d\times d}$ of elements in the bases $\mathcal{B}$. For instance, $\mathrm{U}\mathbf{b}_k$ and $\mathrm{V}\mathbf{b}_k$ represent the generated key and value embeddings. For a query to the neural dictionary, it first computes the similarity (a.k.a. the attention) with all keys ($\mathrm{U}\mathbf{b}_k$), and the corresponding output of the query is the attention-weighted combination of all the elements in the value set ($\mathrm{V}\mathbf{b}_k$).

In a "fake" K-shot N-way few-shot task from $\mathcal{S}$, there are $\mathcal{C}$ categories. Denote $\mathbb{I}\left[\,\mathbf{y}_i = c\,\right]$ as an indicator that selects instances in the class c. To synthesize classifier for a class c, we first compute the class signature as the embedding



(a) Classifier Synthesis using Few-Shot Training Data

(b) Inference with Generalized Few-Shot Learning

Fig. 3 Illustration of Adaptive ClAssifier SynThesize LEarning (ACASTLE). A neural dictionary contains two types of neural bases—the shared component and the task-specific component. During the inference, both the prototype of the tail classes and the descriptors of the SEEN classifier are input into the neural dictionary, and synthesize the joint classifier over both SEEN and UNSEEN categories

prototype, defined as the average embedding of all K shots of instances (in a K-shot N-way task):[3]

$$\mathbf{p}_c = \frac{1}{K} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_{\mathbf{train}}} \phi(\mathbf{x}_i) \cdot \mathbb{I}[\mathbf{y}_i = c] \tag{6}$$

The specific component in the neural dictionary bases $\mathcal{B}$ is the concatenation of the prototype of few-shot instances $\{\mathbf{p}_c\}$ and the linear classifier descriptors set $\mathbf{\Theta}_{\mathcal{S}-\mathcal{C}}$ over the embedding ϕ, i.e.,

$$\mathcal{B}_{\mathbf{specific}} = \{\mathbf{p}_c \mid c \in \mathcal{C}\} \cup \mathbf{\Theta}_{\mathcal{S}-\mathcal{C}} \tag{7}$$

We then compute the attention coefficients α_c for assembling the classifier of class c, via measuring the compatibility score between the class signature and the key embeddings of the neural dictionary,

$$\alpha_c^k \propto \exp\left(\mathbf{p}_c^\top \mathrm{U}\mathbf{b}_k\right), \text{where } k = 1, \ldots, |\mathcal{B}|$$

The coefficient α_c^k is then *normalized* with the sum of compatibility scores over all $|\mathcal{B}|$ bases, which then is used to convexly combine the value embeddings and synthesize the classifier,

$$\mathbf{w}_c = \mathbf{p}_c + \sum_{k=1}^{|\mathcal{B}|} \alpha_c^k \cdot \mathrm{V}\mathbf{b}_k \tag{8}$$

We formulate the classifier composition as a summation of the initial prototype embedding $\mathbf{p}_c$ and the residual component $\sum_{k=1}^{|\mathcal{B}|} \alpha_c^k \cdot \mathrm{V}\mathbf{b}_k$. Such a composed classifier is then ℓ_2-normalized and used for (generalized) few-shot classification. Such normalization also fixes the scale differences in the concatenation of the prototype and the descriptors set in the specific neural bases in Eq. 7. The same classifier synthesis process could be applied to the elements in the SEEN class descriptors set $\mathbf{\Theta}_{\mathcal{S}-\mathcal{C}}$, where a head classifier first computes its similarity with the shared neural bases and the tail prototypes, then adapts the classifier to $\hat{\mathbf{\Theta}}_{\mathcal{S}-\mathcal{C}}$ with Eq. 8. Therefore, the SEEN classifier is also synthesized conditioned on the context of the UNSEEN instances, which promotes the backward knowledge transfer from UNSEEN classes to the SEEN ones.

Since both the embedding "key" and classifier "value" are generated based on the same set of neural bases, it encodes a compact set of latent features for a wide range of classes in $\mathcal{B}_{\mathbf{share}}$ while leaving the task-specific characteristic in $\mathcal{B}_{\mathbf{specific}}$. We hope the learned neural bases contain a rich set of classifier primitives to be transferred to novel compositions of emerging visual categories. Figure 3 demonstrates the classifier synthesize process with the neural dictionary.

We denote the degenerated version with only the shared neural bases $\mathcal{B} = \mathcal{B}_{\mathbf{share}}$ as CASTLE, which makes a joint prediction with the stationary many-shot classifier $\mathbf{\Theta}_{\mathcal{S}}$ and the synthesized few-shot classifier.

Remark 1 Changpinyo et al. (2016, 2020) take advantage of the dictionary to synthesize the classifier for all classes in zero-shot learning. Gidaris and Komodakis (2018) implement a GFSL model with two stages. After pre-training a many-shot classifier, it freezes the embedding and composes the tail classifier by convex combinations of the transforms of the head classifier. Different from the previous approach constructing a dictionary based on a pre-fixed feature embedding, we use a *learned* embedding function together with the neural dictionary, leading to an end-to-end GFSL framework. Furthermore, different from Gidaris and Komodakis (2018) keeping the head classifier stationary, we adapt them conditioned on the tail classes, which could handle the diversity

[3] More choices of Eq. 6 are investigated in "Appendix 9".

between class domains (as illustrated in Fig. 2). Comprehensive experiments to verify the effectiveness of such an adaptive GFSL classifier could be found in Sects. 5 and 6.3.

Remark 2 The attention mechanism to synthesize the classifier is similar to Vaswani et al. (2017), which is also verified to be effective for adapting embeddings for few-shot learning (Ye et al. 2020). Different from Vaswani et al. (2017), both the specific and shared weights are included in the "key" and "value" part of the neural dictionary. No additional normalization strategies (e.g., layer normalization (Ba et al. 2016) and temperature scaling (Guo et al. 2017)) are used in our module.

4 Experimental Setups

This section details the experimental setups, including the general data splits strategy, the pre-training technique, the specifications of the feature backbone, and the evaluation metrics for GFSL.

4.1 Data Splits

We visualize the general data split strategy in Fig. 4. There are two parts of the dataset for standard meta-learning tasks. The meta-training set for model learning (corresponds to the SEEN classes), and the meta-val/test part for model evaluation (corresponds to the UNSEEN classes). To evaluate a GFSL model, we'd like to augment the meta-training set with new instances, so that the classification performance on SEEN classes could be measured. During the inference phase, a few-shot training set from UNSEEN classes are provided with the model, and the model should make a joint prediction over instances from *both* the head and tail classes. We will describe the detailed splits for particular datasets in later sections.

4.2 Pre-training Strategy

Before the meta-training stage, we try to find a good initialization for the embedding ϕ, and then we reuse such a many-shot classifier as well as the embedding to facilitate the training of a GFSL model. More details of the pre-training stage could be found in "Appendix 1". In later sections, we will verify this pre-training strategy does not influence the few-shot classification performance a lot, but it is essential to make the GFSL classifier well-calibrated.

4.3 Feature Network Specification

Following the setting of most recent methods (Qiao et al. 2018; Rusu et al. 2019; Ye et al. 2020), we use ResNet variants (He et al. 2016; Bertinetto et al. 2019) to implement the embedding backbone ϕ.[4] Details of the architecture and the optimization strategy are in "Appendix 2".

4.4 Evaluation Measures

We take advantage of the auxiliary meta-training set from the benchmark datasets during GFSL evaluations, and an illustration of the dataset construction can be found in Fig. 4. The notation $X \rightarrow Y$ with $X, Y \in \{\mathcal{S}, \mathcal{U}, \mathcal{S} \cup \mathcal{U}\}$ means computing prediction results with instances from X to labels of Y. For example, $\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$ means we first filter instances come from the SEEN class set ($\mathbf{x} \in \mathcal{S}$), and predict them into the joint label space ($\mathbf{y} \in \mathcal{S} \cup \mathcal{U}$). For a GFSL model, we consider its performance with different measurements.

4.4.1 Few-Shot Accuracy

Following the standard protocol (Vinyals et al. 2016; Finn et al. 2017; Snell et al. 2017; Ye et al. 2020), we sample 10,000 K-shot N-way tasks from $\mathcal{U}$ during inference. In detail, we first sample N classes from $\mathcal{U}$, and then sample $K + 15$ instances for each class. The first NK labeled instances (K instances from each of the N classes) are used to build the few-shot classifier, and the remaining $15N$ (15 instances from each of the N classes) are used to evaluate the quality of such few-shot classifier. During our test, we consider $K = 1$ and $K = 5$ as in the literature, and change N ranges from $\{5, 10, 15, \ldots, |\mathcal{U}|\}$ as a more robust measure. It is noteworthy that in this test stage, all the instances come from $\mathcal{U}$ and are predicted to classes in $\mathcal{U}$ ($\mathcal{U} \rightarrow \mathcal{U}$).

4.4.2 Generalized Few-Shot Accuracy

Different from many-shot and few-shot evaluations, the generalized few-shot learning takes the joint instance and label spaces into consideration. In other words, the instances come from $\mathcal{S} \cup \mathcal{U}$ and their predicted labels also in $\mathcal{S} \cup \mathcal{U}$ ($\mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$). This is obviously more difficult than the many-shot ($\mathcal{S} \rightarrow \mathcal{S}$) and few-shot ($\mathcal{U} \rightarrow \mathcal{U}$) tasks. During the test, with a bit abuse of notations, we sample K-shot $\mathcal{S} + N$-way tasks from $\mathcal{S} \cup \mathcal{U}$. Concretely, we first sample a K-shot N-way task from $\mathcal{U}$, with NK training and $15N$ test instances, respectively. Then, we *randomly* sample $15N$ instances from $\mathcal{S}$. Thus in a GFSL evaluation task, there are NK labeled instances from $\mathcal{U}$, and $30N$ test instances from $\mathcal{S} \cup \mathcal{U}$. We compute the accuracy of $\mathcal{S} \cup \mathcal{U}$ as the final measure. We abbreviate this criterion as "*Mean Acc.*" or "*Acc.*" in later sections.

[4] Our implementation will be publicly available on https://github.com/Sha-Lab/aCASTLE.

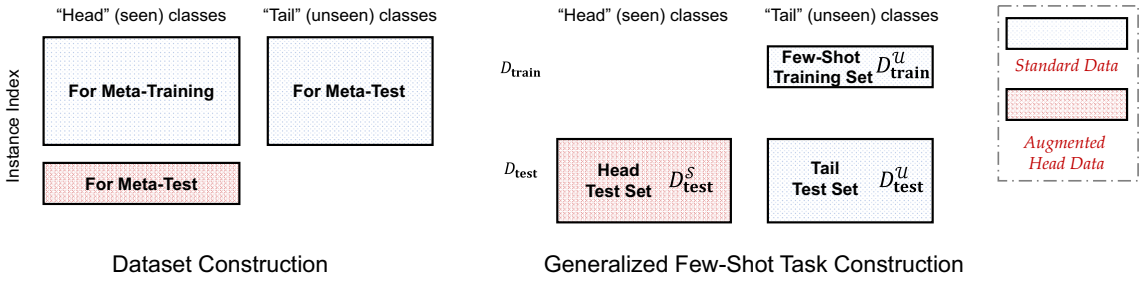


Fig. 4 The split of data in the generalized few-shot classification scenario. In addition to the standard dataset like *Mini*Imagetnet (blue part), we collect non-overlapping augmented head class instances from the corresponding categories in the ImageNet (red part), to measure the classification ability on the SEEN classes. Then in the generalized few-shot classification task, few-shot instances are sampled from each of the UNSEEN classes, while the model should have the ability to predict instances from *both* the head and tail classes (Color figure online)

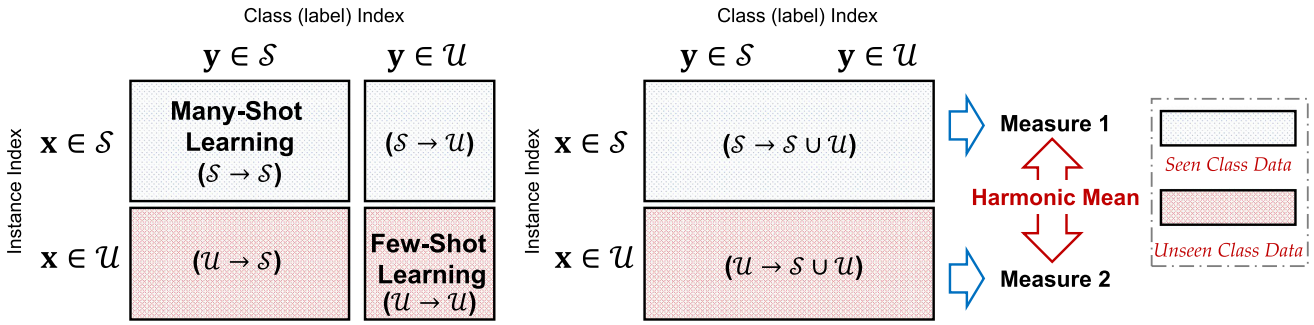


Fig. 5 An illustration of the harmonic mean based criterion for GFSL evaluation. $\mathcal{S}$ and $\mathcal{U}$ denotes the SEEN and UNSEEN instances ($\mathbf{x}$) and labels ($\mathbf{y}$) respectively. $\mathcal{S} \cup \mathcal{U}$ is the joint set of $\mathcal{S}$ and $\mathcal{U}$. The notation $X \to Y, X, Y \in \{\mathcal{S}, \mathcal{U}, \mathcal{S} \cup \mathcal{U}\}$ means computing prediction results with instances from X to labels of Y. By computing a performance measure (like accuracy) on the joint label space prediction of SEEN and UNSEEN instances separately, a harmonic mean is computed to obtain the final measure

4.4.3 Generalized Few-Shot Δ-Value

Since the problem becomes difficult when the predicted label space expands from $\mathcal{S} \to \mathcal{S}$ to $\mathcal{S} \to \mathcal{S} \cup \mathcal{U}$ (and also $\mathcal{U} \to \mathcal{U}$ to $\mathcal{U} \to \mathcal{S} \cup \mathcal{U}$), the accuracy of a model will have a drop. To measure how the classification ability of a GFSL model changes when working in a GFSL scenario, Ren et al. (2019) propose the Δ-Value to measure the average accuracy drop. In detail, for each sampled GFSL task, we first compute its many-shot accuracy ($\mathcal{S} \to \mathcal{S}$) and few-shot accuracy ($\mathcal{U} \to \mathcal{U}$). Then we calculate the corresponding accuracy of SEEN and UNSEEN instances in the joint label space, i.e., $\mathcal{S} \to \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \to \mathcal{S} \cup \mathcal{U}$. The Δ-Value is the average decrease of accuracy in these two cases. We abbreviate this criterion as "Δ-value" in later sections.

4.4.4 Generalized Few-Shot Harmonic Mean

Directly computing the accuracy still gets biased towards the populated classes, so we also consider the harmonic mean as a more balanced measure (Xian et al. 2017). By computing performance measurement such as top-1 accuracy for $\mathcal{S} \to \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \to \mathcal{S} \cup \mathcal{U}$, the harmonic mean is used to average the performance in these two cases as the final measure. In other words, denote the accuracy for $\mathcal{S} \to \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \to \mathcal{S} \cup \mathcal{U}$ as $\text{Acc}_{\mathcal{S}}$ and $\text{Acc}_{\mathcal{U}}$, respectively, the value $\frac{2\text{Acc}_{\mathcal{S}}\text{Acc}_{\mathcal{U}}}{\text{Acc}_{\mathcal{S}}+\text{Acc}_{\mathcal{U}}}$ is used as a final measure. An illustration is in Fig. 5. We abbreviate this criterion as "*HM*" or "*HM Acc.*" in later sections.

4.4.5 Generalized Few-Shot AUSUC

Chao et al. (2016) propose a calibration-agnostic criterion for generalized zero-shot learning. To avoid evaluating a model influenced by a calibration factor between SEEN and UNSEEN classes, they propose to determine the range of the calibration factor for all instances at first, and then plot the SEEN–UNSEEN accuracy curve based on different configurations of the calibration values. Finally, the area under the SEEN–UNSEEN curve is used as a more robust criterion. We follow Chao et al. (2016) to compute the AUSUC value for sampled GFSL tasks. We abbreviate this criterion as "AUSUC" in later sections.

5 Pivot Study on Multi-domain GFSL

We first present a pivot study to demonstrate the effectiveness of ACASTLE, which leverages adaptive classifiers synthesized for both SEEN and UNSEEN classes. To achieve this, we investigate two multi-domain datasets—"Heterogeneous" and "Office-Home" with more challenging settings, where a GFSL model is required to *transfer knowledge in backward direction* (adapt SEEN classifiers based on UNSEEN ones) to obtain superior joint classification performances over heterogeneous domains.

5.1 Dataset

We construct a **Heterogeneous** dataset based on 5 fine-grained classification datasets, namely AirCraft (Maji et al. 2013), Car-196 (Krause et al. 2013), Caltech-UCSD Birds (CUB) 200-2011 (Wah et al. 2011), Stanford Dog (Khosla et al. 2011), and Indoor Scenes (Quattoni and Torralba 2009). Since these datasets have apparent heterogeneous semantics, we treat images from different datasets as different domains. 20 classes with 50 images in each of them are randomly sampled from each of the 5 datasets to construct the meta-training set. The same sampling strategy is also used to sample classes for model validation (meta-val) and evaluation (meta-test) sets. Therefore, there are 100 classes for meta-training/val/test sets, which contains 20 classes from each fine-grained dataset. To evaluate the performance of a GFSL model, we augment the meta-training set by sampling another 15 images from the corresponding classes for each of the SEEN classes.

We also investigate the **Office-Home** (Venkateswara et al. 2017) dataset, which originates from a domain adaptation task. There are 65 classes and 4 domains of images per class. Considering the scarce number of images in one particular domain, we select three of the four domains, "Clipart", "Product", and "Real World" to construct our dataset. The number of instances in a class per domain is not equal. We randomly sample 25 classes (with all selected domains) for meta-training, 15 classes for meta-validation, and the remaining 25 classes are used for meta-test. Similarly, we hold out 10 images per domain for each SEEN class to evaluate the generalized classification ability of a GFSL model.

Note that in addition to the class label, images in these two datasets are also equipped with *at least one* domain label. In particular, classes in Heterogeneous dataset belong to a single domain corresponding to "aircraft", "bird", "car", "dog", or "indoor scene", while the classes in Office-Home possess images from all 3 domains, namely "Clipart", "Product" and "Real World". An illustration of the sampled images (of different domains) from these two datasets is shown in Fig. 6.

The key difference to standard GFSL (cf. Sect. 6.3) is that here the SEEN categories are collected from multiple (heterogeneous) visual domains and used for training the inductive GFSL model. During the evaluation, the few-shot training instances of tail classes *only come from one single domain*. With this key difference, we note that the UNSEEN few-shot classes are close to a certain sub-domain of SEEN classes and relatively far away from the others. Therefore, a model capable of adapting its SEEN classifiers can take the advantages and adapt itself to the domain of the UNSEEN classes.

5.2 Baselines and Comparison Methods

Besides CASTLE and ACASTLE, we consider two other baseline models. The first one optimizes the Eq. 5 directly but without the neural dictionary, which relies on both the (fixed) linear classifier $\mathbf{\Theta}_{\mathcal{S}}$ and the few-shot prototypes to make a GFSL prediction (we denote it as "CASTLE^{-}"); the second one is DFSL (Gidaris and Komodakis 2018), which requires a two-stage training of the GFSL model. It trains a many-shot classifier with cosine similarity in the first stage. Then it freezes the backbone model as feature extractor and optimizes a similar form of Eq. 5 via composing new few-shot classifiers as the convex combination of those many-shot classifiers. It can be viewed as a degenerated neural dictionary, where DFSL sets a size-$|\mathcal{S}|$ "shared" bases $\mathcal{B}_{\mathbf{share}}$ as the many-shot classifier $\mathbf{\Theta}_{\mathcal{S}}$. We observe that DFSL is unstable to perform end-to-end learning. It is potentially because the few-shot classifier composition uses many-shot classifiers as bases, but those bases are optimized to both be good bases and good classifiers, which can likely to be conflicting to some degree. It is also worth noting that all the baselines except ACASTLE only modify the few-shot classifiers, and it is impossible for them to perform backward knowledge transfer.

5.3 GFSL over Heterogeneous Dataset

The Heterogeneous dataset has 100 SEEN classes in the meta-training set, 20 per domain. We consider the case where during the inference, all of the tail classes come from one particular domain. For example, the tail classes are different kinds of birds, and we need to do a joint classification over all SEEN classes from the heterogeneous domains and the newly coming tail classes with limited instances. To mimic the inference scenario, we sample "fake" few-shot tasks with classes from one of the five domains randomly and contrasting the discerning ability from the sampled classes w.r.t. the remaining SEEN classes as in Eq. 5.

Note that we train DFSL strictly follows the strategy in Gidaris and Komodakis (2018), and train other GFSL models with a pre-trained embedding and the multi-classifier techniques to improve the training efficiency. Following Xian et al. (2017), Schönfeld et al. (2019) and Gidaris and Komodakis (2018), we compute the 1-Shot 5-Way GFSL

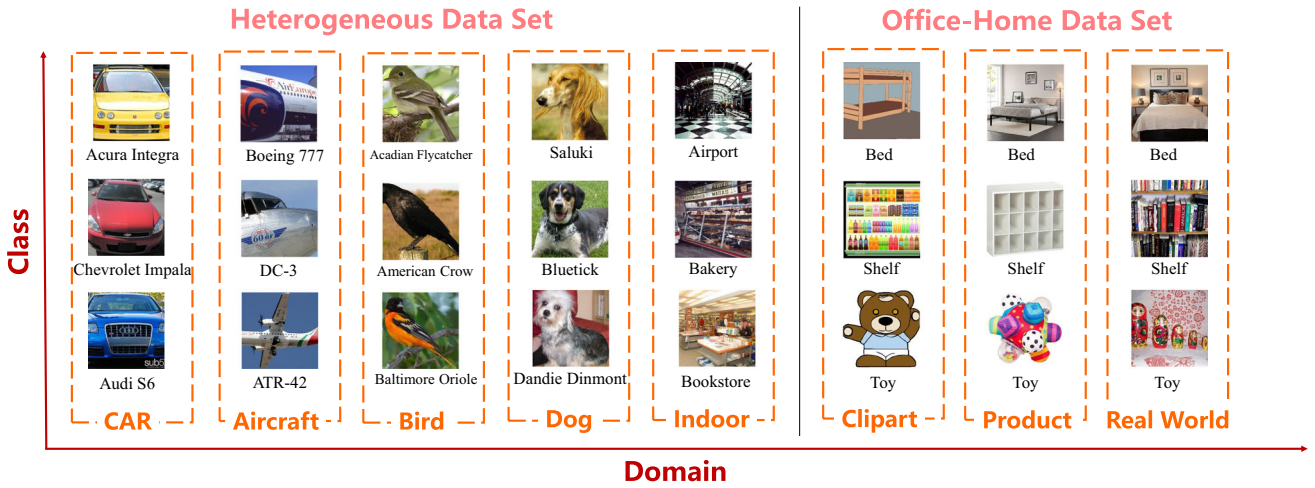


Fig. 6 An illustration of the Heterogeneous and Office-Home dataset. Both datasets contain multiple domains. In the Heterogeneous dataset, each class belongs to only one domain, while in Office-Home, a class has images from all three domains

Table 1 Generalized 1-shot classification performance (mean accuracy and harmonic mean accuracy) on (a) the Heterogeneous dataset with 100 Head and 5 Tail categories and (b) the Office-Home dataset with 25 Head and 5 Tail categories

Measures	$\mathcal{S} \cup \mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$	$\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$	$\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$	*HM Acc.*
(a) Heterogeneous dataset				
DFSL (Gidaris and Komodakis 2018)	48.13 ± 0.12	46.33 ± 0.12	48.25 ± 0.22	47.27 ± 0.12
CASTLE$^-$	48.29 ± 0.12	45.13 ± 0.13	50.14 ± 0.22	47.50 ± 0.12
CASTLE	50.16 ± 0.13	48.05 ± 0.13	$\mathbf{50.86} \pm 0.22$	49.05 ± 0.12
ACASTLE	$\mathbf{53.01} \pm 0.12$	$\mathbf{56.18} \pm 0.12$	49.84 ± 0.22	$\mathbf{52.81} \pm 0.13$
(b) Office–Home dataset				
DFSL (Gidaris and Komodakis 2018)	35.72 ± 0.12	28.42 ± 0.12	39.77 ± 0.22	33.15 ± 0.12
CASTLE$^-$	35.74 ± 0.13	27.93 ± 0.13	$\mathbf{42.59} \pm 0.22$	33.73 ± 0.13
CASTLE	35.77 ± 0.13	29.03 ± 0.13	42.46 ± 0.22	34.48 ± 0.13
ACASTLE	$\mathbf{39.99} \pm 0.14$	$\mathbf{40.29} \pm 0.13$	39.68 ± 0.22	$\mathbf{39.98} \pm 0.14$

$\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$ denote the joint classification accuracy for SEEN class and UNSEEN class instances respectively. CASTLE$^-$ is the variant of CASTLE without using the neural dictionary

The highest mean accuracy and the highest harmonic mean accuracy are in bold

classification mean accuracy and harmonic mean accuracy over 10,000 sampled tasks, whose results are recorded in Table 1a. $\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$ denote the average accuracy for the joint prediction of SEEN and UNSEEN instances respectively.

From the results in Table 1a, DFSL could not work well due to its fixed embedding and restricted bases. CASTLE$^-$ is able to balance the training accuracy of both SEEN and UNSEEN classes benefited from the pre-train strategy and the unified learning objective, which achieves the highest joint classification performance over UNSEEN classes. The discriminative ability is further improved with the help of the neural dictionary. CASTLE performs better than its degenerated version, which verifies the effectiveness of the learned neural bases. The neural dictionary encodes the common characteristic among all classes for the GFSL classification, so that CASTLE gets better mean accuracy and harmonic mean accuracy than CASTLE$^-$. Since ACASTLE is able to adapt both many-shot and few-shot classifiers conditioned on the context of the tail instances, it obtains the best GFSL performance in this case. It is notable that ACASTLE gets much higher joint classification accuracy for SEEN classes than other methods, which validates its ability to adapt the many-shot classifier over the SEEN classes based on the context of tail classes.

5.4 GFSL over Office-Home Dataset

We also investigate the similar multi-domain GFSL classification task over the Office-Home dataset. However, in this case, a single class could belong to all three domains. We consider the scenario to classify classes in a single domain and

the domain of the classes should be inferred from the limited tail instances. In other words, we train a GFSL model over 25 classes, and each class has 3 sets of instances corresponding to the three domains. In meta-training, a 25-way SEEN class classifier is constructed. During the inference, the model is provided by another 5-way 1-shot set of UNSEEN class instances from one single domain. The model is required to output a joint classifier for test instances from the whole 30 classes whose domains are the same as the one in the UNSEEN class set.

Towards such a multi-domain GFSL task, we train a GFSL model by keeping the instances in both the few-shot fake tail task and corresponding test set from the same domain. We use the same set of comparison methods and evaluation protocols with the previous subsection. The mean accuracy, harmonic mean accuracy, and the specific accuracy for SEEN and UNSEEN classes are shown in Table 1b.

Due to the ambiguity of domains for each class, the GFSL classification over Office-Home gives rise to a more difficult problem, while the results in Table 1b reveal a similar trend with those in Table 1a. Since for Office-Home a single GFSL model needs to make the joint prediction over classes from multiple domains conditioned on different configurations of the tail few-shot tasks, the stationary SEEN class classifiers are not suitable for the classification over different domains. In this case, ACASTLE still achieves the best performance over different GFSL criteria, and gets larger superiority margins with the comparison methods.

6 Experiments on GFSL

In this section, we design experiments on benchmark datasets to validate the effectiveness of the CASTLE and ACASTLE in GFSL (cf. Sect. 6.3). After a comprehensive comparison with competitive methods using various protocols, we analyze different aspects of GFSL approaches, and we observe the post calibration makes the FSL methods strong GFSL baselines. We verify that CASTLE/ACASTLE learn a better calibration between SEEN and UNSEEN classifiers, and the neural dictionary makes CASTLE/ACASTLE persist its high discerning ability with incremental tail few-shot instances. Finally, we show that CASTLE/ACASTLE also benefit standard FSL performances (cf. Sect. 6.4).

6.1 Datasets

Two benchmark datasets are used in our experiments. The *Mini***ImageNet** dataset (Vinyals et al. 2016) is a subset of the ILSVRC-12 dataset (Russakovsky et al. 2015). There are totally 100 classes and 600 examples in each class. For evaluation, we follow the split of Ravi and Larochelle (2017) and use 64 of 100 classes for meta-training, 16 for validation, and 20 for meta-test (model evaluation). In other words, a model is trained on few-shot tasks sampled from the 64 SEEN classes set during meta-training, and the best model is selected based on the few-shot classification performance over the 16 class set. The final model is evaluated based on few-shot tasks sampled from the 20 UNSEEN classes.

The *Tiered***ImageNet** (Ren et al. 2018) is a more complicated version compared with the *Mini*ImageNet. It contains 34 super-categories in total, with 20 for meta-training, 6 for validation (meta-val), and 8 for model testing (meta-test). Each of the super-category has 10 to 30 classes. In detail, there are 351, 97, and 160 classes for meta-training, meta-validation, and meta-test, respectively. The divergence of the super-concept leads to a more difficult few-shot classification problem.

Since both datasets are constructed by images from ILSVRC-12, we augment the *meta-training* set of each dataset by sampling non-overlapping images from the corresponding classes in ILSVRC-12. The auxiliary meta-train set is used to measure the generalized few-shot learning classification performance on the SEEN class set. For example, for each of the 64 SEEN classes in the *Mini*ImageNet, we collect 200 more non-overlapping images per class from ILSVRC-12 as the test set for many-shot classification. The illustration of the dataset split is shown in Fig. 4.

6.2 Baselines and Prior Methods

We explore several (strong) choices in deriving classifiers for the SEEN and UNSEEN classes, including Multiclass Classifier (MC) + kNN, which contains a $|\mathcal{S}|$-way classifier trained on the SEEN classes in a supervised learning manner as standard many-shot classification, and its embedding with the nearest neighbor classifier is used for GFSL inference; ProtoNet + ProtoNet, where the embeddings trained by Prototypical Network (Snell et al. 2017) is used, and 100 training instances are sampled from each SEEN category to act as the SEEN class prototypes; MC + ProtoNet, where we combine the learning objective of the previous two baselines to jointly learn the MC classifier and feature embedding. Details of the methods are in "Appendix 3".

Besides, we also compare our approach with the L2ML (Wang et al. 2017), Dynamic Few-Shot Learning without forgetting (**DFSL**) (Gidaris and Komodakis 2018), and the newly proposed Incremental few-shot learning (**IFSL**) (Ren et al. 2019). For CASTLE, we use the many-shot classifiers $\{\boldsymbol{\Theta}_{\mathcal{S}}\}$, cf. Sect. 3.1) for the SEEN classes and the synthesized classifiers for the UNSEEN classes to classify an instance into all classes, and then select the prediction with the highest confidence score. For ACASTLE, we adapt the head classifiers to $\{\hat{\boldsymbol{\Theta}}_{\mathcal{S}}\}$ with the help of the tail classes.

Table 2 Generalized Few-shot classification performance (mean accuracy, Δ-value, and harmonic mean accuracy) on *Mini*ImageNet when there are **64 Head and 5 Tail categories**

Setups	1-Shot		5-Shot		1-Shot	5-Shot
Perf. Measures	*Mean Acc.* ↑	Δ ↓	*Mean Acc.*↑	Δ ↓	*Harmonic Mean Acc.*↑	
IFSL (Ren et al. 2019)	54.95 ± 0.30	11.84	63.04 ± 0.30	10.66	–	–
L2ML (Wang et al. 2017)	46.25 ± 0.04	27.49	45.81 ± 0.03	35.53	2.98 ± 0.06	1.12 ± 0.04
DFSL (Gidaris and Komodakis 2018)	63.36 ± 0.11	13.71	72.58 ± 0.09	13.33	62.08 ± 0.13	71.26 ± 0.09
MC + *k*NN	46.17 ± 0.03	29.70	46.18 ± 0.03	40.21	0.00 ± 0.00	0.00 ± 0.00
MC + ProtoNet	45.31 ± 0.03	29.71	45.85 ± 0.03	39.82	0.00 ± 0.00	0.00 ± 0.00
ProtoNet + ProtoNet	50.49 ± 0.08	25.64	71.75 ± 0.08	13.65	19.26 ± 0.18	67.73 ± 0.12
Ours: CASTLE	67.13 ± 0.11	10.09	76.78 ± 0.09	9.88	66.22 ± 0.15	76.32 ± 0.09
Ours: ACASTLE	**68.70** ± 0.11	**9.98**	**78.63** ± 0.09	**8.08**	**66.24** ± 0.15	**78.33** ± 0.09

The highest mean accuracy, highest harmonic mean accuracy, and the lowest Δ-value are in bold

6.3 Main Results

We first evaluate all GFSL methods on *Mini*ImageNet with the criteria in Gidaris and Komodakis (2018) and Ren et al. (2019), the mean accuracy over all classes (the higher the better) and the Δ-value (the lower the better). An effective GFSL approach not only makes prediction well on the joint label space (with high accuracy) but also keeps its classification ability when changing from many-shot/few-shot to the generalized few-shot case (low Δ-value).

The main results are shown in Table 2. We found that ACASTLE outperforms all the existing methods as well as our proposed baseline systems in terms of the mean accuracy. Meanwhile, when looked at the Δ-value, and CASTLE variants are the least affected between predicting for SEEN/USSEEN classes separately and predicting over all classes jointly.

However, we find that either mean accuracy or Δ-value is not informative enough to tell about a GFSL algorithm's performances. For example, a baseline system, i.e., ProtoNet + ProtoNet performs better than IFSL in terms of 5-shot mean accuracy but not Δ-value. This is consistent with the observation in Ren et al. (2019) that the Δ-value should be considered together with the mean accuracy. *In this case, how shall we rank these two systems?* To answer this question, we propose to use another evaluation measure, the harmonic mean of the mean accuracy for each SEEN and UNSEEN category (Xian et al. 2017; Schönfeld et al. 2019), when they are classified jointly.

6.3.1 Harmonic Mean Accuracy Measures GFSL Performance Better

Since the number of SEEN and UNSEEN classes are most likely to be not equal, e.g., 64 versus 5 in our cases, directly computing the mean accuracy over all classes is almost always biased. For example, a many-shot classifier that only classifies samples into SEEN classes can receive a good performance than one that recognizes both SEEN and UNSEEN. Therefore, we argue that *harmonic mean* over the mean accuracy can better assess a classifier's performance, as now the performances are negatively affected when a classifier ignores classes (e.g., MC classifier get 0% harmonic mean). Specifically, we compute the top-1 accuracy for instances from SEEN and UNSEEN classes, and take their harmonic mean as the performance measure. The results are included in the right side of the Table 2.

We find the harmonic mean accuracy takes a holistic consideration of the "absolute" joint classification performance and the "relative" performance drop when classifying towards the joint set. For example, the many-shot baseline MC+*k*NN with good mean accuracy and high Δ-value has extremely low performance as it tends to ignore UNSEEN categories. Meanwhile, CASTLE and ACASTLE remain the best when ranked by the harmonic mean accuracy against others.

6.3.2 Evaluate GFSL Beyond 5 UNSEEN Categories

Besides using harmonic mean accuracy, we argue that another important aspect in evaluating GFSL is to go beyond the 5 sampled UNSEEN categories, as it is never the case in real-world. On the contrary, we care most about the GFSL with a large number of UNSEEN classes, which also measure the ability of the model to extrapolate the number of novel classes in the UNSEEN class few-shot task. To this end, we consider an extreme case—evaluating GFSL with *all available* SEEN and UNSEEN categories over both *Mini*ImageNet and *Tiered*ImageNet, and report their results in Tables 3 and 4.

Together with the harmonic mean accuracy of *all* categories, we also report the tail classification performance, which is a more challenging few-shot classification task (the standard FSL results could be found in Sect. 6.4). In addition, the joint classification accuracy for SEEN classes instances

Table 3 Generalized Few-shot classification accuracies on *Mini*ImageNet with 64 head categories and 20 tail categories

Classification on	20 UNSEEN Categories		64 SEEN + 20 UNSEEN Categories					
Perf. Measures	$\mathcal{U} \rightarrow \mathcal{U}$		$\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$		$\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$		*HM Acc.*	
Setups	1-Shot	5-Shot	1-Shot	5-Shot	1-Shot	5-Shot	1-Shot	5-Shot
L2ML (Wang et al. 2017)	27.79 ± 0.07	43.42 ± 0.06	90.99 ± 0.03	90.99 ± 0.03	0.64 ± 0.00	1.21 ± 0.01	1.27 ± 0.09	2.38 ± 0.02
DFSL (Gidaris and Komodakis 2018)	33.02 ± 0.08	50.96 ± 0.07	61.68 ± 0.06	66.06 ± 0.05	**31.13 ± 0.07**	**47.16 ± 0.06**	41.21 ± 0.07	54.95 ± 0.05
MC + *k*NN	31.58 ± 0.08	56.08 ± 0.06	**92.35 ± 0.03**	92.38 ± 0.03	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
MC + ProtoNet	31.82 ± 0.06	56.16 ± 0.06	91.39 ± 0.03	**92.99 ± 0.03**	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
ProtoNet + ProtoNet	32.90 ± 0.08	55.69 ± 0.06	89.15 ± 0.04	85.17 ± 0.04	9.89 ± 0.05	41.17 ± 0.06	17.71 ± 0.08	55.51 ± 0.06
Ours CASTLE	35.69 ± 0.08	56.97 ± 0.06	80.32 ± 0.06	80.43 ± 0.06	29.42 ± 0.08	42.55 ± 0.05	43.06 ± 0.07	55.65 ± 0.07
Ours ACASTLE	**36.38 ± 0.08**	**57.29 ± 0.06**	81.36 ± 0.05	87.40 ± 0.04	29.95 ± 0.08	41.64 ± 0.06	**43.63 ± 0.08**	**56.33 ± 0.06**

The highest mean accuracy and the highest harmonic mean accuracy are in bold

Table 4 Generalized Few-shot classification accuracy on *Tiered*ImageNet with 351 head categories and 160 tail categories

Classification on	160 UNSEEN Categories		351 SEEN + 160 UNSEEN Categories					
Perf. Measures	$\mathcal{U} \rightarrow \mathcal{U}$		$\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$		$\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$		*HM Acc.*	
Setups	1-Shot	5-Shot	1-Shot	5-Shot	1-Shot	5-Shot	1-Shot	5-Shot
DFSL (Gidaris and Komodakis 2018)	15.79 ± 0.02	30.69 ± 0.02	11.29 ± 0.05	14.95 ± 0.06	14.24 ± 0.06	27.22 ± 0.07	12.60 ± 0.11	19.29 ± 0.05
MC + *k*NN	14.12 ± 0.02	30.02 ± 0.02	68.32 ± 0.02	**68.33 ± 0.02**	0.00 ± 0.00	0.00 ± 0.00	0.01 ± 0.00	0.01 ± 0.00
MC + ProtoNet	14.13 ± 0.02	30.05 ± 0.02	**68.34 ± 0.02**	**68.33 ± 0.02**	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
ProtoNet + ProtoNet	14.52 ± 0.02	29.38 ± 0.02	62.37 ± 0.02	61.15 ± 0.02	4.83 ± 0.03	22.69 ± 0.02	8.97 ± 0.02	33.09 ± 0.02
Ours: CASTLE	15.97 ± 0.02	30.44 ± 0.02	26.94 ± 0.08	34.98 ± 0.02	**16.17 ± 0.06**	31.61 ± 0.05	20.20 ± 0.05	33.20 ± 0.02
Ours: ACASTLE	**16.36 ± 0.02**	**30.75 ± 0.02**	27.01 ± 0.08	35.41 ± 0.08	**16.17 ± 0.06**	**31.86 ± 0.05**	**22.23 ± 0.05**	**33.54 ± 0.02**

The highest mean accuracy and the highest harmonic mean accuracy are in bold

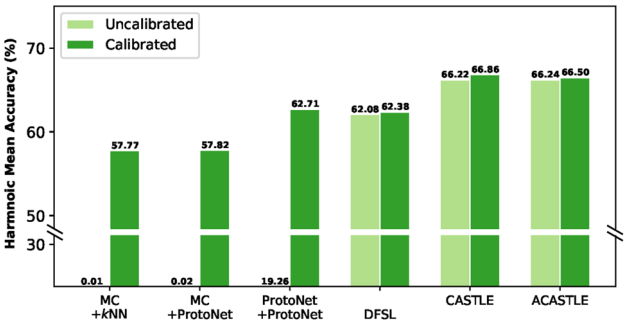


Fig. 7 Calibration's effect to the 1-shot harmonic mean accuracy on *Mini*ImageNet. Baseline models improve a lot with the help of the calibration factor

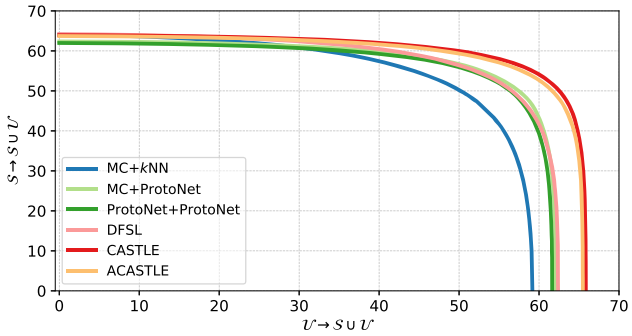


Fig. 8 The 1-shot AUSUC performance with two configurations of UNSEEN classes on *Mini*ImageNet. The larger the area under the curve, the better the GFSL ability.

($\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$) and UNSEEN classes instances ($\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$) are also listed.

The methods without a clear consideration of head-tail trade-off (e.g., ProtoNet+ProtoNet) fails to make a joint prediction over both SEEN and UNSEEN classes. We observe that CASTLE and ACASTLE outperform all approaches in the UNSEEN and more importantly, the ALL categories section, across two datasets.

6.3.3 Confidence Calibration Matters in GFSL

In generalized zero-shot learning, Chao et al. (2016) have identified a significant prediction bias between classification confidence of SEEN and UNSEEN classifiers. We find a similar phenomena in GFSL. For instance, the few-shot learning *ProtoNet + ProtoNet* baseline becomes too confident to predict on SEEN categories than UNSEEN categories (The scale of confidence is on average 2.1 times higher). To address this issue, we compute a calibration factor based on the meta-validation set of UNSEEN categories, such that the prediction logits are calibrated by subtracting this factor out from the confidence of SEEN categories' predictions. With 5 UNSEEN classes from *Mini*ImageNet, the GFSL results of all comparison methods before and after calibration is shown in Fig. 7. We observe a consistent and obvious improvements over the harmonic mean accuracy for all methods. For example, although the FSL approach ProtoNet neglects the classification performance over SEEN categories outside the sampled task during meta-learning, it gets even better harmonic mean accuracy compared with the GFSL method DFSL (62.70% vs. 62.38%) with such post-calibration, which *becomes a very strong GFSL baseline*. Note that CASTLE and ACASTLE are the least affected with the selected calibration factor. This suggests that CASTLE variants, learned with the unified GFSL objective, have well-calibrated classification confidence and does not require additional data and extra learning phase to search this calibration factor.

Moreover, we use area under SEEN–UNSEEN curve (AUSUC) as a measure of different GFSL algorithms (Chao et al. 2016). Here, AUSUC is a performance measure that takes the effects of the calibration factor out. To do so, we enumerate through a large range of calibration factors and subtract it from the confidence score of SEEN classifiers. Through this process, the joint prediction performances over SEEN and UNSEEN categories, denoted as $\mathcal{S} \rightarrow \mathcal{S} \cup \mathcal{U}$ and $\mathcal{U} \rightarrow \mathcal{S} \cup \mathcal{U}$, shall vary as the calibration factor changes. For instance, when the calibration factor is infinitely large, we measure a classifier that only predicts UNSEEN categories. We denote this as the SEEN–UNSEEN curve. The 1-shot GFSL results with 5 UNSEEN classes from *Mini*ImageNet is shown in Fig. 8. As a result, we observe that ACASTLE and CASTLE archive the largest area under the curve, which indicates that CASTLE variants are in general a better algorithm over others among different calibration factors.

6.3.4 Robust Evaluation of GFSL

Other than the harmonic mean accuracy of all SEEN and UNSEEN categories shown in Tables 3 and 4, we study the dynamic of how harmonic mean accuracy changes with an incremental number of UNSEEN tail concepts. In other words, we show the GFSL performances w.r.t. different numbers of tail concepts. We use this as a *robust evaluation* of each system's GFSL capability. In addition to the test instances from the head 64 classes in *Mini*ImageNet, 5 to 20 novel classes are included to compose the generalized few-shot tasks. Concretely, only one instance per novel class is used to construct the tail classifier, combined with which the model is asked to do a *joint* classification of both SEEN and UNSEEN classes. Figure 9 records the change of generalized few-shot learning performance (harmonic mean) when more UNSEEN classes emerge. We omit the results of MC+*k*NN and MC+ProtoNet since they bias towards SEEN classes and get nearly zero harmonic mean accuracy in all cases. We observe that ACASTLE consistently outperforms all baseline

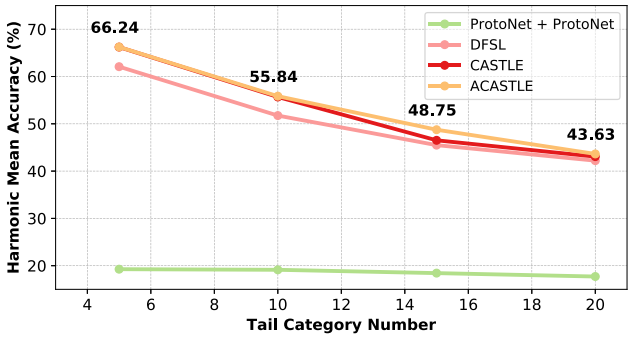


Fig. 9 Results of 1-shot GFSL harmonic mean accuracy with incremental number of UNSEEN classes on *Mini*ImageNet. Note MC+*k*NN and MC+ProtoNet bias towards SEEN classes and get nearly zero harmonic mean accuracy

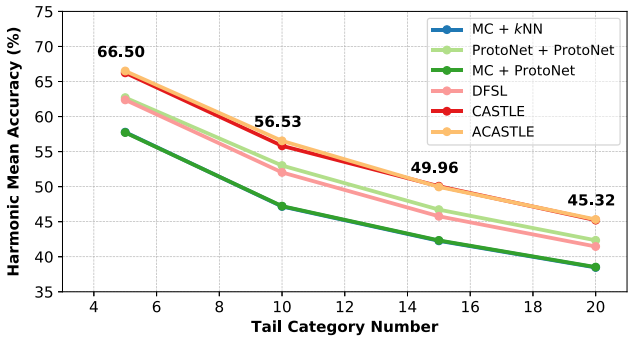


Fig. 10 *Post-calibrated* results of 1-shot GFSL harmonic mean accuracy with incremental number of UNSEEN classes on *Mini*ImageNet. All methods select the their best calibration factors from the meta-val data split

approaches in each evaluation setup, with a clear margin. We also compute the harmonic mean after selecting the best calibration factor from the meta-val set (cf. Fig. 10). It is obvious that almost all baseline models achieve improvements and the phenomenon is consistent with Fig. 7. The GFSL results of ACASTLE and CASTLE are almost not influenced after using the post-calibration technique. ACASTLE still persists its superiority in this case.

6.4 Standard Few-Shot Learning

Finally, we also evaluate our proposed approaches' performance on two standard few-shot learning benchmarks, i.e., *Mini*ImageNet and *Tiered*ImageNet dataset. In other words, we evaluate the classification performance of few-shot UNSEEN class instances with our GFSL objective. We compare our approaches with the state-of-the-art methods in both 1-shot 5-way and 5-shot 5-way scenarios. We cite the results of the comparison methods from their published papers and remark the backbones used to train the FSL model by different methods. The mean accuracy and 95% confidence interval are shown in the Table 5 and Table 6.

It is notable that some comparison methods such as CTM Li et al. (2019) are evaluated over only 600 UNSEEN class FSL tasks, while we test both CASTLE and ACASTLE over 10,000 tasks, leading to more stable results. CASTLE and ACASTLE achieve almost the best 1-shot and 5-shot classification results on both datasets. The results support our hypothesis that jointly learning with many-shot classification forces few-shot classifiers to be discriminative.

7 Related Work and Discussion

Building a high-quality visual system usually requires to have a large scale of annotated training set with many shots per category. Many large-scale datasets such as ImageNet have an ample number of instances for popular classes (Russakovsky et al. 2015; Krizhevsky et al. 2017). However, the data-scarce tail of the category distribution matters. For example, a visual search engine needs to deal with the rare object of interests (e.g., endangered species) or newly defined items (e.g., new smartphone models), which only possesses a few data instances. Directly training a system over all classes is prone to over-fit and can be biased towards the data-rich categories (Cui et al. 2019; Cao et al. 2019; Kang et al. 2020; Ye et al. 2020; Zhou et al. 2020).

Zero-shot learning (ZSL) (Lampert et al. 2014; Akata et al. 2013; Xian et al. 2017; Changpinyo et al. 2020) is a popular idea for addressing learning without labeled data. By aligning the visual and semantic definitions of objects, ZSL transfers the relationship between images and attributes learned from SEEN classes to UNSEEN ones, so as to recognize a novel instance with only its category-wise attributes (Changpinyo et al. 2016, 2017). Generalized ZSL (Chao et al. 2016; Schönfeld et al. 2019) extends this by calibrating a prediction bias to jointly predict between SEEN and UNSEEN classes. ZSL is limited to recognizing objects with well-defined semantic descriptions, which assumes that the visual appearance of novel categories is harder to obtain than knowledge about their attributes, whereas in the real-world we often get the appearance of objects before learning about their characteristics.

Few-shot learning (FSL) proposes a more realistic setup, where we have access to a very limited number (instead of zero) of visual exemplars from the tail classes (Li et al. 2006; Vinyals et al. 2016). FSL meta-learns an inductive bias from the SEEN classes, such that it transfers to the learning process of UNSEEN classes with few training data during the model deployment. For example, one line of works uses meta-learned discriminative feature embeddings (Snell et al. 2017; Oreshkin et al. 2018; Rusu et al. 2019; Vuorio et al. 2019; Lee et al. 2019; Ye et al. 2020) together with the non-parametric nearest neighbor classifiers, to recognize novel classes given a few exemplars. Another line of works chooses to learn the

Table 5 Few-shot classification accuracy on *Mini*ImageNet with different types of backbones

Setups	Backbone	**1-Shot 5-Way**	**5-Shot 5-Way**
IFSL (Ren et al. 2019)	ResNet-10	55.72 ± 0.41	70.50 ± 0.36
DFSL (Gidaris and Komodakis 2018)	ResNet-10	56.20 ± 0.86	73.00 ± 0.64
ProtoNet (Snell et al. 2017)	ResNet-12	61.40 ± 0.12	76.56 ± 0.20
TapNet (Yoon et al. 2019)	ResNet-12	61.65 ± 0.15	76.36 ± 0.10
MTL (Sun et al. 2019)	ResNet-12	61.20 ± 1.80	75.50 ± 0.90
MetaOptNet (Lee et al. 2019)	ResNet-12	62.64 ± 0.61	78.63 ± 0.46
FEAT (Ye et al. 2020)	ResNet-12	66.78 ± 0.20	82.05 ± 0.14
SimpleShot (Wang et al. 2019)	ResNet-18	62.85 ± 0.20	80.02 ± 0.14
CTM (Li et al. 2019)	ResNet-18	64.12 ± 0.82	80.51 ± 0.13
LEO (Rusu et al. 2019)	WRN	61.76 ± 0.08	77.59 ± 0.12
Ours: CASTLE	ResNet-12	66.75 ± 0.20	81.98 ± 0.14
Ours: ACASTLE	ResNet-12	**66.83** ± 0.20	**82.08 ± 0.14**

Our methods are evaluated with 10,000 few-shot tasks
The highest mean accuracy values are in bold

Table 6 Few-shot classification accuracy on *Tiered*ImageNet with different types of backbones

Setups	Backbone	**1-Shot 5-Way**	**5-Shot 5-Way**
ProtoNet (Snell et al. 2017)	ConvNet	53.31 ± 0.89	72.69 ± 0.74
IFSL (Ren et al. 2019)	ResNet-18	51.12 ± 0.45	66.40 ± 0.36
DFSL (Gidaris and Komodakis 2018)	ResNet-18	50.90 ± 0.46	66.69 ± 0.36
TapNet (Yoon et al. 2019)	ResNet-12	63.08 ± 0.15	80.26 ± 0.12
MTL (Sun et al. 2019)	ResNet-12	65.60 ± 1.80	78.60 ± 0.90
MetaOptNet (Lee et al. 2019)	ResNet-12	65.99 ± 0.72	81.56 ± 0.63
FEAT (Ye et al. 2020)	ResNet-12	70.80 ± 0.23	84.79 ± 0.16
SimpleShot (Wang et al. 2019)	ResNet-18	69.09 ± 0.22	84.58 ± 0.16
CTM (Li et al. 2019)	ResNet-18	68.41 ± 0.39	84.28 ± 1.73
LEO (Rusu et al. 2019)	WRN	66.33 ± 0.05	81.44 ± 0.09
Ours: CASTLE	ResNet-12	71.14 ± 0.02	84.34 ± 0.16
Ours: ACASTLE	ResNet-12	**71.63** ± 0.02	**85.28 ± 0.15**

Our methods are evaluated with 10,000 few-shot tasks
The highest mean accuracy values are in bold

common optimization strategy (Ravi and Larochelle 2017; Bertinetto et al. 2019) across few-shot tasks, e.g., the model initialization to a pre-specified model configuration could be adapted rapidly using fixed steps of gradient descents over the few-shot training data from UNSEEN categories (Finn et al. 2017; Li et al. 2017; Nichol et al. 2018; Lee et al. 2018; Antoniou et al. 2019). FSL has achieved promising results in various domains such as visual recognition (Triantafillou et al. 2017; Lifchitz et al. 2019; Das and Lee 2020), domain adaptation (Dong and Xing 2018; Kang et al. 2018), neural machine translation (Gu et al. 2018), data compression (Wang et al. 2018), and density estimation (Reed et al. 2018). Empirical studies of FSL could be found in (Chen et al. 2019; Triantafillou et al. 2020).

FSL emphasizes on building models of the UNSEEN classes, while the simultaneous recognition of the many-shot head categories in real-world use cases is also important. Low-shot learning has been studied in this manner (Hariharan and Girshick 2017; Wang et al. 2018; Gao et al. 2018; Ye et al. 2020; Liu et al. 2019). The main aim is to recognize the entire set of concepts in a transductive learning framework—during the training of the target model, it has access to both the (many-shot) SEEN and (few-shot) UNSEEN categories. The key difference with our Generalized Few-Shot Learning (GFSL) is that we assume no access to UNSEEN classes in the model learning phase, which requires the model to *inductively* transfer knowledge from SEEN classes to UNSEEN ones during the model evaluation phase.

Some of the previous GFSL approaches (Hariharan and Girshick 2017; Wang et al. 2018; Gao et al. 2018) apply the exemplar-based classification paradigms on both SEEN and UNSEEN categories to resolve the transductive learning

problem, which requires recomputing the centroids for SEEN categories after model updates. Others (Wang et al. 2017; Schönfeld et al. 2019; Liu et al. 2019) usually ignore the explicit relationship between SEEN and UNSEEN categories, and learn separate classifiers. Ren et al. (2019) and Gidaris and Komodakis (2018) propose to solve inductive GFSL via either composing UNSEEN with SEEN classifiers or meta-leaning with recurrent back-propagation procedure. Gidaris and Komodakis (2018) is the most related work to CASTLE and ACASTLE, which composes the tail classifiers by a convex combination of the many-shot classifiers. CASTLE is different from Gidaris and Komodakis (2018) as it presents an *end-to-end learnable framework* with improved training techniques, as well as it employs *a shared neural dictionary* to compose few-shot classifiers. Moreover, ACASTLE further relates the knowledge for both SEEN and UNSEEN classes by constructing a neural dictionary with both shared (yet task-agnostic) and task-specific basis, which allows backward knowledge transfer to benefit SEEN classifiers with new knowledge of UNSEEN classes. As we have demonstrated in Sect. 5, ACASTLE significantly improves SEEN classifiers when learning of UNSEEN visual categories over heterogeneous visual domains.

8 Conclusion

A Generalized Few-Shot Learning (GFSL) model takes both the discriminative ability of many-shot and few-shot classifiers into account. In this paper, we propose the ClAssifier SynThesis LEarning (CASTLE) and its adaptive variant (ACASTLE) to solve the challenging inductive modeling of UNSEEN tail categories in conjunction with seen head ones. Our approach takes advantage of the neural dictionary to learn bases for composing many-shot and few-shot classifiers via a unified learning objective, which transfers the knowledge from SEEN to UNSEEN classifiers better. Our experiments highlight ACASTLE especially fits the GFSL scenario with tasks from multiple domains. Both CASTLE and ACASTLE not only outperform existing methods in terms of various GFSL criteria but also improve the classifier's discernibility over standard FSL. Future directions include improving the architecture of neural dictionary and designing better fine-tuning strategies for GFSL.

Acknowledgements Thanks to Fei Sha for valuable discussions. This research (61773198, 61751306, 61632004, 62006112), NSFC-NRF Joint Research Project under Grant 61861146001, NSF Awards IIS-1513966/ 1632803/1833137, CCF-1139148, DARPA Award#: FA8750-18-2-0117, DARPA-D3M - Award UCB-00009528, Google Research Awards, gifts from Facebook and Netflix, and ARO# W911NF-12-1-0241 and W911NF-15-1-0484.

Appendix A: Implementation Details

A.1 Pre-training Strategy

In particular, on *Mini*ImageNet, we add a linear layer on the backbone output and optimize a 64-way classification problem on the meta-training set with the cross-entropy loss function. Stochastic gradient descent with initial learning rate 0.1 and momentum 0.9 is used to complete such optimization. The 16 classes in *Mini*ImageNet for model selection also assist the choice of the pre-trained model. After each epoch, we use the current embedding and measures the nearest neighbor based few-shot classification performance on the sampled few-shot tasks from these 16 classes. The most suitable embedding function is recorded. After that, such a learned backbone is used to initialize the embedding part ϕ of the whole model. The same strategy is also applied to the meta-training set of the *Tiered*ImageNet, Heterogeneous, and Office-Home datasets, where a 351-way, 100-way, and 25-way classifiers are pre-trained.

A.2 Feature Network Specification

We follow Qiao et al. (2018); Rusu et al. (2019) when investigating the multi-domain GFSL, where images are resized to $84 \times 84 \times 3$. In concrete words, three residual blocks are used after an initial convolutional layer (with stride 1 and padding 1) over the image, which have channels 160/320/640, stride 2, and padding 2. After a global average pooling layer, it leads to a 640 dimensional embedding. While for the benchmark experiments on *Mini*ImageNet and *Tiered*ImageNet, we follow Lee et al. (2019) to set the architecture of ResNet, which contains 12 layers and uses the DropBlock (Ghiasi et al. 2018) to prevent over-fitting.

We use the pre-trained backbone to initialize the embedding part ϕ of a model for CASTLE/ACASTLE and our re-implemented comparison methods such as MC+kNN, ProtoNet+ProtoNet, MC+ProtoNet, L2ML (Wang et al. 2017), and DFSL (Gidaris and Komodakis 2018). When there exists a backbone initialization, we set the initial learning rate as 1e-4 and optimize the model with Momentum SGD. The learning rate will be halved after optimizing 2,000 mini-batches. During meta-learning, all methods are optimized over 5-way few-shot tasks, where the number of shots in a task is consistent with the inference (meta-test) stage. For example, if the goal is a 1-shot 5-way model, we sample 1-shot 5-way $\mathcal{D}^{\mathcal{S}}_{\mathbf{train}}$ during meta-training, together with 15 instances per class in $\mathcal{D}^{\mathcal{S}}_{\mathbf{test}}$.

For CASTLE/ACASTLE, we take advantage of the multi-classifier training technique to improve learning efficiency. We randomly sample a 24-way task from $\mathcal{S}$ in each mini-batch, and re-sample 64 5-way tasks from it. It is notable that all instances in the 24-way task are encoded by the ResNet

backbone with the same parameters in advance. Therefore, by embedding the synthesized 5-way few-shot classifiers into the global many-shot classifier, it results in 64 different configurations of the generalized few-shot classifiers. To evaluate the classifier, we randomly sample instances with batch size 128 from $\mathcal{S}$ and compute the GFSL objective in Eq. 5.

A.3 Baselines for GFSL Benchmarks

Here we describe some baseline approaches compared in the GFSL benchmarks in detail.

(1) Multiclass Classifier (MC) + kNN A $|\mathcal{S}|$-way classifier is trained on the SEEN classes in a supervised learning manner as standard many-shot classification (He et al. 2016). During the inference, test examples of $\mathcal{S}$ categories are evaluated based on the $|\mathcal{S}|$-way classifiers and $|\mathcal{U}|$ categories are evaluated using the support embeddings from $\mathcal{D}^{\mathcal{U}}_{\textbf{train}}$ with a nearest neighbor classifier. To evaluate the generalized few-shot classification task, we take the union of multi-class classifiers' confidence and nearest neighbor confidence [the normalized negative distance values as in Snell et al. (2017)] as joint classification scores on $\mathcal{S} \cup \mathcal{U}$.

(2) ProtoNet + ProtoNet We train a few-shot classifier (initialized by the MC classifier's feature mapping) using the Prototypical Network (Snell et al. 2017) (a.k.a. ProtoNet), pretending they were few-shot. When evaluated on the SEEN categories, we randomly sample 100 training instances per category to compute the class prototypes. The class prototypes of UNSEEN classes are computed based on the sampled few-shot training set. During the inference of *generalized* few-shot learning, the confidence of a test instances is jointly determined by its (negative) distance to both SEEN and UNSEEN class prototypes.

(3) MC + ProtoNet We combine the learning objective of the previous two baselines ((1) and (2)) to jointly learn the MC classifier and feature embedding. Since there are two objectives for many-shot (cross-entropy loss on all SEEN classes) and few-shot (ProtoNet meta-learning objective) classification respectively, it trades off between many-shot and few-shot learning. Therefore, this learned model can be used as multi-class linear classifiers on the head categories, and used as ProtoNet on the tail categories. During the inference, the model predicts instances from SEEN class $\mathcal{S}$ with the MC classifier, while takes advantage of the few-shot prototypes to discern UNSEEN class instances. To evaluate the generalized few-shot classification task, we take the union of multi-class classifiers' confidence and ProtoNet confidence as joint classification scores on $\mathcal{S} \cup \mathcal{U}$.

(4) L2ML Wang et al. (2017) propose learning to model the "tail" (L2ML) by connecting a few-shot classifier with the corresponding many-shot classifier. The method is designed to learn classifier dynamics from data-poor "tail" classes to the data-rich head classes. Since L2ML is originally designed to learn with both SEEN and UNSEEN classes in a transductive manner. In our experiment, we adaptive it to our setting. Therefore, we learn a classifier mapping based on the sampled few-shot tasks from SEEN class set $\mathcal{S}$, which transforms a few-shot classifier in UNSEEN class set $\mathcal{U}$ inductively. Following Wang et al. (2017), we first train a many-shot classifier W upon the ResNet backbone on the SEEN class set $\mathcal{S}$. We use the same residual architecture as in Wang et al. (2017) to implement the classifier mapping f, which transforms a few-shot classifier to a many-shot classifier. During the meta-learning stage, a $\mathcal{S}$-way few-shot task is sampled in each mini-batch, which produces a $\mathcal{S}$-way linear few-shot classifier $\hat{W}$ based on the fixed pre-trained embedding. The objective of L2ML not only regresses the mapped few-shot classifier $f(\hat{W})$ close to the many-shot one W measured by square loss, but also minimizes the classification loss of $f(\hat{W})$ over a randomly sampled instances from $\mathcal{S}$. Therefore, L2ML uses a pre-trained multi-class classifier W for those head categories and used the predicted few-shot classifiers with f for the tail categories.

Appendix B: More Analysis on GFSL Benchmarks

In this appendix, we do analyses to show the influence of training a GFSL model by reusing the many-shot classifier and study different implementation choices in the proposed methods. We mainly investigate and provide the results over CASTLE on *Mini*ImageNet. We observe the results on ACASTLE and other datasets reveal similar trends.

B.1 Reusing the Many-Shot Classifier Facilitates the Calibration for GFSL

We compare the strategy to train CASTLE from scratch and fine-tune based on the many-shot classifier. We show both the results of 1-Shot 5-Way few-shot classification performance and GFSL performance with 5 UNSEEN tasks for CASTLE when trained from random or with provided initialization. From the results in Table 7, we find training from scratch gets only a bit lower few-shot classification results with the fine-tune strategy, but much lower GFSL harmonic mean accuracy. Therefore, reusing the parameters in the many-shot classifier benefits the predictions on SEEN and UNSEEN classes of a GFSL model. Therefore, we use the pre-trained embedding to initialize the backbone.

Table 7 The difference between training with a pre-trained backbone or from scratch with 1-Shot 5-Way Tasks on *Mini*ImageNet

Perf. Measures	FSL *MA*	GFSL *HM*
CASTLE w/ pre-train	66.83 ± 0.21	66.22 ± 0.15
CASTLE w/o pre-train	64.23 ± 0.21	38.24 ± 0.09

MA mean accuracy and *HM* harmonic mean accuracy

B.2 Comparison with One-Phase Incremental Learning Methods

The inductive generalized few-shot learning is also related to the one-phase incremental learning (Li and Hoiem 2018; Liu et al. 2020), where a model is required to adapt itself to the open set environment. In other words, after training over the closed set categories, a classifier should be updated based on the data with novel distributions or categories accordingly. One important thread of incremental learning methods relies on the experience replay, where a set of the closed set instances is preserved and the classifier for all classes is optimized based on the saved and novel few-shot data. In our inductive GFSL, the CASTLE variants do not save SEEN class instances and rely on the neural dictionary to adapt the classifier for a joint classification. Thus, CASTLE variants have lower computational (time) costs during the inference stage.

Towards comprehensive comparisons, we also investigate two popular incremental learning methods, i.e., LwF (Li and Hoiem 2018) and iCARL (Li and Hoiem 2018). We randomly save 5 images per SEEN class for both methods. By combining the stored images and the newly given UNSEEN class images together, the model will be updated based on a cross-entropy loss and a distillation loss (Hinton et al. 2015). We tune the balance weight between the classification and distillation loss, the initial learning rate for fine-tuning, and the optimization steps for both methods over the validation set. The harmonic mean accuracy in various evaluation scenarios over 10,000 tasks are listed in Table 8.

In our empirical evaluations, we find that incremental learning methods can get better results than our baselines since it fine-tunes the model with the distillation loss. However, their results are not stable since there are many hyper-parameters. Compared with these approaches, our CASTLE variants still keep their superiority over all criteria.

B.3 Light-Weight Adaptation on CASTLE Variants

As shown in the previous subsection, directly fine-tuning the whole model is prone to over-fit even with another distillation loss. Inspired by Sun et al. (2019) and Li et al. (2019), we consider a light-weight fine-tune step based on the synthesized classifier by CASTLE variants. In detail, we reformulate the model $\mathbf{W}^\top \phi(\mathbf{x})$ as $\mathbf{W}^\top((\mathbf{1}+\text{scale}) \cdot \phi(\mathbf{x}) + \text{bias})$, where $\mathbf{W}$ is the classifier output by the neural dictionary, the scale $\in \mathbb{R}^d$ and bias $\in \mathbb{R}^d$ are additional learnable vectors, and $\mathbf{1}$ is a size d vector with all values equal 1.

Given a few-shot task with UNSEEN class instances, the model will be updated in the following ways. 5 images per SEEN class are randomly selected, after freezing the backbone ϕ, the classifier $\mathbf{W}$, the scale, and the bias are optimized based on a cross-entropy loss over both stored SEEN and UNSEEN classes images. We tune the initial learning rate and the optimization steps over the validation set.

The results of such model adaptation strategies are listed in Table 9. With further model adaptation, both CASTLE and ACASTLE could be improved.

B.4 Effects on the Neural Dictionary Size $|\mathcal{B}|$

We show the effects of the dictionary size (as the ratio of SEEN class size 64) for the standard few-shot learning (measured by mean accuracy when there are 5 UNSEEN classes) in Fig. 11. We observe that the neural dictionary with a ratio of 2 or 3 works best amongst all other dictionary sizes. Therefore, we fix the dictionary size as 128 across all experiments. Note that when $|\mathcal{B}| = 0$, our method degenerates to case optimizing the unified objective in Eq. 5 without using the neural dictionary (the CASTLE$^-$ model in Sect. 5).

B.5 How Well is Synthesized Classifiers Comparing with Multi-class Classifiers?

To assess the quality of synthesized classifier, we made a comparison against ProtoNet and also the Multi-class Classifier on the head SEEN concepts. To do so, we sample few-shot

Table 8 Comparison between CASTLE variants and the incremental learning methods on *Mini*ImageNet

Classification on	5-Way		20-Way	
Setups	1-Shot	5-Shot	1-Shot	5-Shot
LwF (Li and Hoiem 2018)	60.18 ± 0.15	73.48 ± 0.09	28.70 ± 0.06	39.88 ± 0.06
iCARL (Li and Hoiem 2018)	61.14 ± 0.15	73.58 ± 0.09	31.60 ± 0.06	46.55 ± 0.06
CASTLE	66.22 ± 0.15	76.32 ± 0.09	43.06 ± 0.07	55.65 ± 0.07
ACASTLE	66.24 ± 0.15	78.33 ± 0.09	43.63 ± 0.08	56.33 ± 0.06

The harmonic mean accuracy in different evaluation scenarios are recorded

Table 9 The light-weight model adaptation by fine-tuning the scale and bias weights based on the classifier initialization from CASTLE variants

Classification on	5-Way		20-Way	
Setups	1-Shot	5-Shot	1-Shot	5-Shot
CASTLE	66.22 ± 0.15	76.32 ± 0.09	43.06 ± 0.07	55.65 ± 0.07
CASTLE†	66.24 ± 0.15	76.43 ± 0.09	43.12 ± 0.07	55.85 ± 0.07
ACASTLE	66.24 ± 0.15	78.33 ± 0.09	43.63 ± 0.08	56.33 ± 0.06
ACASTLE†	66.33 ± 0.15	78.93 ± 0.09	43.68 ± 0.08	56.42 ± 0.06

The harmonic mean accuracy in different evaluation scenarios on *Mini*ImageNet are recorded. The superscript † denotes the method with another light-weight update step

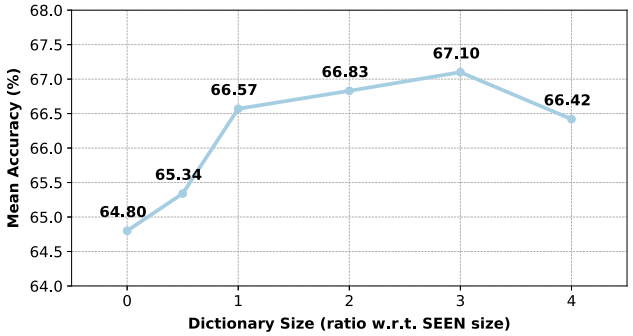


Fig. 11 The 1-shot 5-way accuracy on UNSEEN of *Mini*ImageNet with different size of dictionaries

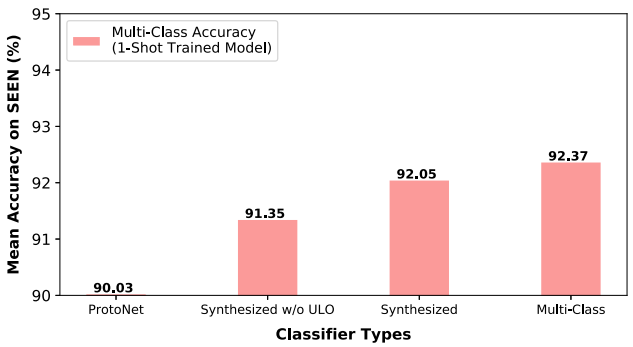


Fig. 12 The 64-way multi-class accuracy on SEEN of *Mini*ImageNet with 1-shot trained model

training instances on each SEEN category to synthesize classifiers (or compute class prototypes for ProtoNet), and then use the synthesized classifiers/class prototypes solely to evaluate multi-class accuracy. The results are shown in Fig. 12. We observe that the learned synthesized classifier outperforms over ProtoNet. Also, the model trained with unified learning objective improves over the vanilla synthesized classifiers. Note that there is still a gap left against multi-class classifiers trained on the entire dataset. It suggests that the classifier synthesis we learned is effective against using sole instance embeddings.

Table 10 The performance with different choices of classifier synthesize strategies when tested with 5-Shot 5-Way UNSEN Tasks on *Mini*ImageNet

Perf. Measures	FSL *Mean Acc.*	GFSL *HM Acc.*
CASTLE w/ Pre-AVG	81.98 ± 0.20	76.32 ± 0.09
CASTLE w/ Post-AVG	82.00 ± 0.20	76.28 ± 0.09

We denote the option compute embedding prototype and average synthesized classifiers as "Pre-AVG" and "Post-AVG" respectively

B.6 Different Choices of the Classifier Synthesis

As in Eq. 6, when there is more than one instance per class in a few-shot task (i.e., $K > 1$), CASTLE compute the averaged embeddings first, and then use the prototype of each class as the input of the neural dictionary to synthesize their corresponding classifiers. Here we explore another choice to deal with multiple instances in each class. We synthesize classifiers based on each instance first, and then average the corresponding synthesized classifiers for each class. This option equals an ensemble strategy to average the prediction results of each instance's synthesized classifier. We denote the pre-average strategy (the one used in CASTLE) as "Pre-AVG", and the post-average strategy as "Post-AVG". The 5-Shot 5-way classification results on *Mini*ImageNet for these two strategies are shown in Table 10. From the results, "Post-AVG" does not improve the FSL and GFSL performance obviously. Since averaging the synthesized classifiers in a hindsight way costs more memory during meta-training, we choose the "Pre-AVG" option to synthesize classifiers when there are more than 1 shot in each class. In our experiments, the same conclusion also applies to ACASTLE.

B.7 How is *Multiple Classifiers Learning's* Impact over the Training?

Both CASTLE and ACASTLE adopt a multi-classifier training strategy (as described in Sect. 3), i.e., considering multiple GFSL tasks with different combinations of classifiers in a single mini-batch. In Table 11, we show the influence of the multi-classifier training method based on their GFSL performance (harmonic mean). It shows that with a large number

Table 11 The GFSL performance (harmonic mean accuracy) change with different number of classifiers (# of CLS) when tested with 1-Shot 5-Way UNSEN Tasks on *Mini*ImageNet

# of Classifiers	1	64	128	256
CASTLE	64.53 ± 0.15	65.61 ± 0.15	66.22 ± 0.15	66.72 ± 0.15

Table 12 The performance gap between CASTLE variants and a kind of "many-shot" upper bound (denoted as "UB") on *Mini*ImageNet

Setups	5-Way		20-Way	
Measures	FSL	GFSL	FSL	GFSL
CASTLE	81.98 ± 0.14	76.32 ± 0.09	56.97 ± 0.06	43.06 ± 0.07
ACASTLE	82.08 ± 0.14	78.33 ± 0.09	57.29 ± 0.06	56.33 ± 0.06
UB	87.08 ± 0.10	80.23 ± 0.09	68.25 ± 0.05	68.72 ± 0.12

The ability of FSL classification is measured by the mean accuracy, while the harmonic mean accuracy is used as a criterion for GFSL. 5-Shot classification performance of CASTLE and ACASTLE are listed for a comparison

of classifiers during the training, the performance of CASTLE asymptotically converges to its upper-bound. We find ACASTLE shares a similar trend.

B.8 The Gap to the Performance "Upper Bound" (UB)

We focus on the (generalized) few-shot learning scenario where there are only budgeted examples in the UNSEEN class tasks. To show the potential improvement space in such tasks, we also investigate a kind of upper bound model where all the available images are used to build the UNSEEN class classifier during the inference stage.

We implement the upper bound model based on the ProtoNet, and the results are in Table 12. Specifically, in the FSL classification scenario, all the UNSEEN class images except those preserved for evaluation are used to build more precise prototypes, and the mean accuracy over 10,000 tasks are recorded; in the GFSL classification scenario, the many-shot UNSEEN class images are utilized as well, and the calibrated harmonic mean is used as the performance measure.

Since the upper bound takes advantage of all the available training images for the few-shot categories, it performs better than the few-shot CASTLE and ACASTLE in all the scenarios. The gap between the few-shot learning methods and the upper bound becomes larger when more UNSEEN classes (ways) are involved.

References

Akata, Z., Perronnin, F., Harchaoui, Z., & Schmid, C. (2013). Label-embedding for attribute-based classification. In *IEEE conference on computer vision and pattern recognition* (pp. 819–826).

Antoniou, A., Edwards, H., & Storkey, A. J. (2019). How to train your MAML. In *Proceedings of the 7th international conference on learning representations*.

Ba, L. J., Kiros, R., & Hinton, G. E. (2016). Layer normalization. CoRR arXiv:1607.06450.

Bertinetto, L., Henriques, J. F., Torr, P. H. S., & Vedaldi, A. (2019). Meta-learning with differentiable closed-form solvers. In *Proceedings of the 7th international conference on learning representations*.

Cao, K., Wei, C., Gaidon, A., Arechiga, N., & Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in Neural Information Processing Systems*, *32*, 1565–1576.

Changpinyo, S., Chao, W. L., & Sha, F. (2017). Predicting visual exemplars of unseen classes for zero-shot learning. In *IEEE international conference on computer vision* (pp. 3496–3505).

Changpinyo, S., Chao, W. L., Gong, B., & Sha, F. (2016). Synthesized classifiers for zero-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 5327–5336).

Changpinyo, S., Chao, W. L., Gong, B., & Sha, F. (2020). Classifier and exemplar synthesis for zero-shot learning. *International Journal of Computer Vision*, *128*(1), 166–201.

Chao, W. L., Changpinyo, S., Gong, B., & Sha, F. (2016). An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *Proceedings of the 14th European conference on computer vision* (pp. 52–68).

Chen, W. Y., Liu, Y. C., Kira, Z., Wang, Y. C. F., & Huang, J. B. (2019). A closer look at few-shot classification. In *Proceedings of the 7th international conference on learning representations*.

Cui, Y., Jia, M., Lin, T. Y., Song, Y., & Belongie, S. J. (2019). Class-balanced loss based on effective number of samples. In *IEEE conference on computer vision and pattern recognition* (pp. 9268–9277).

Das, D., & Lee, C. S. G. (2020). A two-stage approach to few-shot learning for image recognition. *IEEE Transactions on Image Processing*, *2*(9), 3336–3350.

Dong, N., & Xing, E. P. (2018). Domain adaption in one-shot learning. In *Proceedings of the European conference on machine learning and knowledge discovery in databases* (pp. 573–588).

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th international conference on machine learning* (pp. 1126–1135).

Gao, H., Shou, Z., Zareian, A., Zhang, H., & Chang, S. F. (2018). Low-shot learning via covariance-preserving adversarial augmentation networks. *Advances in Neural Information Processing Systems*, *31*, 983–993.

Ghiasi, G., Lin, T. Y., & Le, Q. V. (2018). Dropblock: A regularization method for convolutional networks. *Advances in Neural Information Processing Systems*, *31*, 10750–10760.

Gidaris, S., & Komodakis, N. (2018). Dynamic few-shot visual learning without forgetting. In *IEEE international conference on computer vision* (pp. 4367–4375).

Gu, J., Wang, Y., Chen, Y., Li, V. O. K., & Cho, K. (2018). Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 3622–3631).

Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th international conference on machine learning* (pp. 1321–1330).

Hariharan, B., & Girshick, R. B. (2017). Low-shot visual recognition by shrinking and hallucinating features. In *IEEE international conference on computer vision* (pp. 3037–3046).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hinton, G. E., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. CoRR arXiv:1503.02531.

Kang, B., & Feng, J. (2018). Transferable meta learning across domains. In *Proceedings of the 34th conference on uncertainty in artificial intelligence* (pp. 177–187).

Kang, B., Xie, S., Rohrbach, M., Yan, Z., Gordo, A., Feng, J., & Kalantidis, Y. (2020). Decoupling representation and classifier for long-tailed recognition. In *Proceedings of the 8th international conference on learning representations*.

Khosla, A., Jayadevaprakash, N., Yao, B., & Fei-Fei, L. (2011). Novel dataset for fine-grained image categorization. In *1st workshop on fine-grained visual categorization, IEEE conference on computer vision and pattern recognition*.

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop* (Vol. 2).

Krause, J., Stark, M., Deng, J., & Fei-Fei, L. (2013). 3D object representations for fine-grained categorization. In *4th international IEEE workshop on 3D representation and recognition*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90.

Lampert, C. H., Nickisch, H., & Harmeling, S. (2014). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *36*(3), 453–465.

Larochelle, H. (2018). Few-shot learning with meta-learning: Progress made and challenges ahead.

Lee, Y., & Choi, S. (2018). Gradient-based meta-learning with learned layerwise metric and subspace. In *Proceedings of the 35th international conference on machine learning* (pp. 2933–2942).

Lee, K., Maji, S., Ravichandran, A., & Soatto, S. (2019). Meta-learning with differentiable convex optimization. In *IEEE conference on computer vision and pattern recognition* (pp. 10657–10665).

Li, H., Eigen, D., Dodge, S., Zeiler, M & Wang, X. (2019). Finding task-relevant features for few-shot learning by category traversal. In *IEEE conference on computer vision and pattern recognition* (pp. 1–10).

Li, Z., Zhou, F., Chen, F., & Li, H. (2017). Meta-SGD: Learning to learn quickly for few shot learning. CoRR arXiv:1707.09835.

Lifchitz, Y., Avrithis, Y., Picard, S., & Bursuc, A. (2019). Dense classification and implanting for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 9258–9267).

Li, F. F., Fergus, R., & Perona, P. (2006). One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(4), 594–611.

Li, Z., & Hoiem, D. (2018). Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *40*(12), 2935–2947.

Li, X., Sun, Q., Liu, Y., Zhou, Q., Zheng, S., Chua, T. S., et al. (2019). Learning to self-train for semi-supervised few-shot classification. *Advances in Neural Information Processing Systems*, *32*, 10276–10286.

Liu, Y., Liu, A. A., Su, Y., Schiele, B., & Sun, Q. (2020). Mnemonics training: Multi-class incremental learning without forgetting. In *IEEE conference on computer vision and pattern recognition* (pp. 12245–12254).

Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., & Yu, S. X. (2019). Large-scale long-tailed recognition in an open world. In *IEEE conference on computer vision and pattern recognition* (pp. 2537–2546).

Lopez-Paz, D., & Ranzato, M. (2017). Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems*, *30*, 6467–6476.

Maji, S., Rahtu, E., Kannala, J., Blaschko, M. B., & Vedaldi, A. (2013). Fine-grained visual classification of aircraft. CoRR arXiv:1306.5151.

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. CoRR arXiv:1803.02999.

Oreshkin, B. N., López, P. R., & Lacoste, A. (2018). TADAM: Task dependent adaptive metric for improved few-shot learning. *Advances in Neural Information Processing Systems*, *31*, 719–729.

Qiao, S., Liu, C., Shen, W., & Yuille, A. L. (2018). Few-shot image recognition by predicting parameters from activations. In *IEEE conference on computer vision and pattern recognition* (pp. 7229–7238).

Quattoni, A., & Torralba, A. (2009). Recognizing indoor scenes. In *IEEE conference on computer vision and pattern recognition* (pp. 413–420).

Ravi, S., & Larochelle, H. (2017). Optimization as a model for few-shot learning. In *Proceedings of the 5th international conference on learning representations*.

Reed, S. E., Chen, Y., Paine, T., van den Oord, A., Eslami, S. M. A., Rezende, D. J., et al. (2018). Few-shot autoregressive density estimation: Towards learning to learn distributions. In *Proceedings of the 6th international conference on learning representations*.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., et al. (2018). Meta-learning for semi-supervised few-shot classification. In *Proceedings of the 6th international conference on learning representations*.

Ren, M., Liao, R., Fetaya, E., & Zemel, R. (2019). Incremental few-shot learning with attention attractor networks. *Advances in Neural Information Processing Systems*, *32*, 5276–5286.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, *115*(3), 211–252.

Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., et al. (2019). Meta-learning with latent embedding optimization. In *Proceedings of the 7th international conference on learning representations*.

Schönfeld, E., Ebrahimi, S., Sinha, S., Darrell, T., & Akata, Z. (2019). Generalized zero- and few-shot learning via aligned variational autoencoders. In *IEEE conference on computer vision and pattern recognition* (pp. 8247–8255).

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd international conference on learning representations*.

Snell, J., Swersky, K., & Zemel, R. S. (2017). Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems*, *30*, 4080–4090.

Sun, Q., Liu, Y., Chua, T. S., & Schiele, B. (2019). Meta-transfer learning for few-shot learning. In *IEEE conference on computer vision and pattern recognition* (pp. 403–412).

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., et al. (2020). Meta-dataset: A dataset of datasets for learning to

learn from few examples. In *Proceedings of the 8th international conference on learning representations*.

Triantafillou, E., Zemel, R. S., & Urtasun, R. (2017). Few-shot learning through an information retrieval lens. *Advances in Neural Information Processing Systems*, *30*, 2252–2262.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *30*, 5998–6008.

Venkateswara, H., Eusebio, J., Chakraborty, S., & Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation. In *IEEE conference on computer vision and pattern recognition* (pp. 5385–5394).

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, *29*, 3630–3638.

Vuorio, R., Sun, S. H., Hu, H., & Lim, J. J. (2019). Multimodal model-agnostic meta-learning via task-aware modulation. *Advances in Neural Information Processing Systems*, *32*, 1–12.

Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). The Caltech-UCSD Birds-200-2011 Dataset. Technical report CNS-TR-2011-001, California Institute of Technology.

Wang, Y., Chao, W. L., Weinberger, K. Q., & van der Maaten, L. (2019). Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. CoRR arXiv:1911.04623.

Wang, Y. X., Girshick, R. B., Hebert, M., & Hariharan, B. (2018). Low-shot learning from imaginary data. In *IEEE conference on computer vision and pattern recognition* (pp. 7278–7286).

Wang, T., Zhu, J. Y., Torralba, A., & Efros, A. A. (2018). Dataset distillation. CoRR arXiv:1811.10959.

Wang, Y. X., Ramanan, D., & Hebert, M. (2017). Learning to model the tail. *Advances in Neural Information Processing Systems*, *30*, 7032–7042.

Xian, Y., Schiele, B., & Akata, Z. (2017). Zero-shot learning—The good, the bad and the ugly. In *IEEE conference on computer vision and pattern recognition* (pp. 3077–3086).

Ye, H. J., Chen, H. Y., Zhan, D. C., & Chao, W. L. (2020). Identifying and compensating for feature deviation in imbalanced deep learning. CoRR arXiv:2001.01385.

Ye, H. J., Hu, H., Zhan, D. C., & Sha, F. (2020). Few-shot learning via embedding adaptation with set-to-set functions. In *IEEE conference on computer vision and pattern recognition* (pp. 8808–8817).

Yoon, S. W., Seo, J., & Moon, J. (2019). Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In *Proceedings of the 36th international conference on machine learning* (pp. 7115–7123).

Zhou, B., Cui, Q., Wei, X. S., & Chen, Z. M. (2020). BBN: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *IEEE conference on computer vision and pattern recognition* (pp. 9719–9728).