Learning to Hallucinate Examples from Extrinsic and Intrinsic Supervision

Liangke Gui\textsuperscript{1}\textsuperscript{*} Adrien Bardes\textsuperscript{2}\textsuperscript{†} Ruslan Salakhutdinov\textsuperscript{1} Alexander Hauptmann\textsuperscript{1} Martial Hebert\textsuperscript{1} Yu-Xiong Wang\textsuperscript{3}
\textsuperscript{1} CMU \textsuperscript{2} Facebook AI Research, Inria \textsuperscript{3} UIUC
{liangkeg, rsalakhu, alex, hebert}@cs.cmu.edu adrien.bardes@inria.fr yxw@illinois.edu

Abstract

Learning to hallucinate additional examples has recently been shown as a promising direction to address few-shot learning tasks. This work investigates two important yet overlooked natural supervision signals for guiding the hallucination process—(i) extrinsic: classifiers trained on hallucinated examples should be close to strong classifiers that would be learned from a large amount of real examples; and (ii) intrinsic: clusters of hallucinated and real examples belonging to the same class should be pulled together, while simultaneously pushing apart clusters of hallucinated and real examples from different classes. We achieve (i) by introducing an additional mentor model on data-abundant base classes for directing the hallucinator, and achieve (ii) by performing contrastive learning between hallucinated and real examples. As a general, model-agnostic framework, our dual mentor- and self-directed (DMAS) hallucinator significantly improves few-shot learning performance on widely-used benchmarks in various scenarios.

1. Introduction

To alleviate the reliance on large, labeled datasets for learning deep models, few-shot learning has attracted increasing attention, with the goal of learning novel concepts from one, or only a few, annotated examples \cite{12, 30, 2}. Existing work tries to solve this problem from the perspective of meta-learning \cite{52, 6, 57}, which is motivated by the human ability to leverage prior experiences when tackling a new task. Unlike the standard machine learning paradigm, where a model is trained on a set of examples, meta-learning is performed on a set of “simulated” tasks, each consisting of its own support and query sets \cite{58}. The support set is used as the few-shot training data for the learner, and the query set is used as the test data to evaluate the learner’s quality. By sampling small support and query sets from a large collection of labeled examples of base classes, meta-learning based approaches learn to extract task-agnostic knowledge, and apply it to a new few-shot learning task of novel classes.

One notable type of task-agnostic (or meta) knowledge comes from the shared mechanism of data augmentation or hallucination across categories \cite{70, 22, 28, 82}. Since synthesizing raw images is often challenging or sometimes unnecessary, recent work has instead focused on hallucinating examples in a learned feature space \cite{70, 22, 28, 82, 81, 86, 87}. This can be achieved by, for example, integrating a “hallucinator” module into a meta-learning framework, where it generates hallucinated examples guided by real ones from the support set \cite{70}. The hallucinator captures the intra-class variation shared across categories, which generalizes to unseen classes. The learner then uses an augmented training set, which includes both the real and the hallucinated examples to...
learn classifiers. The hallucinator is meta-trained end-to-end with the learner, through back-propagating a classification loss based on ground-truth labels of query data.

Despite the success of prior approaches, we argue that solely using the classification loss on the small query set as supervision is insufficient to adjust the hallucinator to produce effective samples in the few-shot regime. Therefore, the performance of the classifiers trained on hallucinated examples is still substantially inferior to that of the classifiers trained on real examples [16, 58]. To overcome this challenge, our key insight is that there are two important yet under-explored natural signals for guiding the data generation process – extrinsic and intrinsic supervision. This work explores how to leverage such supervision to enable hallucinating examples in a way that helps the classification algorithm learn better classifiers.

The first source of supervision is an extrinsic signal from large-sample learning. As illustrated in Figure 1, to be most helpful as a hallucinator, a classifier trained on the hallucinated examples (which are generated from a small support set of real samples) is expected to be close to a strong classifier that would be trained on a large amount of real examples. This extrinsic signal from large-sample learning is a natural source of supervision for few-shot learning, but it has been largely overlooked in prior work. While we have very little data on novel classes, we do have a large number of real examples on base classes. Therefore, on base classes we introduce a “mentor” model, which is a strong classifier pre-trained on all the available large amount of real examples. Correspondingly, the classifier trained on hallucinated examples along with few real support examples becomes the “student.”

We now minimize the discrepancy between the student and mentor classifiers. A straightforward approach would be minimizing the distance between the two classifiers in the parameter space [71, 72, 11], which tends to be difficult and noisy due to the lack of suitable metrics. Hence, we instead encourage the output predictions from the student classifier (e.g., the distribution of class probabilities) to be similar to those predicted by the mentor on the query set. This way of learning is reminiscent of knowledge distillation [29]. By doing so, the hallucinator explicitly learns how to produce examples that enable the student classifier to mimic the behavior of the mentor. Note that the student-mentor pairs are only used for meta-training on base classes; there are no mentor classifiers for meta-testing on novel classes.

In practice, the student and mentor classifiers could be quite different from each other at the beginning of the training, if the mentor is produced by a large amount of real examples while the student has access to only few real examples. To address this issue, we propose a progressive guidance scheme inspired by curriculum learning [8], and explore two dual directions – (1) we start with a mentor and a student, both trained on a small number of real examples, and we gradually strengthen the mentor by re-training it with increasing number of real examples; and (2) we start with a mentor and a student, both trained on a large number of real examples, and we gradually weaken the student by removing its real examples. During both of the processes, the hallucinator is also trained progressively.

The second source of supervision is the intrinsic label consistency between hallucinated and real examples. As illustrated in Figure 1, hallucinated and real examples belonging to the same class should be pulled together, while simultaneously pushing apart clusters of hallucinated and real examples from different classes. However, without appropriate constraints, the hallucinated examples might be noisy and spread over across class boundaries (e.g., a hallucinated dog example resides within the cat cluster). To this end, we formulate the problem as supervised contrastive learning, inspired by recent progress on self-supervised learning [74, 28, 12, 30]. We treat hallucinated and real examples as different views of the data, and generate the positive and negative pairs correspondingly. For example, the positives are drawn from both hallucinated and real samples of the same class. Note that different from conventional contrastive learning that learns an embedding space (where the data augmentation is pre-defined), we use the contrastive loss to self-direct the hallucinated examples in the right class cluster or manifold (where is the feature space is pre-trained).

Our contributions are three-fold. (1) By jointly leveraging the complementary extrinsic and intrinsic supervision, we develop a general meta-learning with hallucination framework. (2) We not only extract shared knowledge across a collection of few-shot learning tasks, similar to most existing meta-learning methods, but also progressively exploit extrinsic knowledge in large-sample models trained on base classes as mentor to guide hallucination and few-shot learning. (3) Through a contrastive learning process, the hallucinated examples are self-directed to maintain the intrinsic label consistency with real examples. Our dual mentor- and self-directed (DMAS) hallucinator is model-agnostic, which can generate data in different feature spaces and can be combined with different classification models to consistently boost their few-shot learning performance on a variety of benchmarks, including ImageNet1K [27, 70], minImageNet [68, 49], tieredImageNet [51], and CUB [69].

2. Related Work

Generative Models. Generative models have recently shown great potential as a way of data augmentation for few-shot learning [5, 70, 84, 22] and semi-supervised learning [16], but the improvement of recognition performance is still limited [58]. The generation can be performed either in image space [15] or in a pre-trained feature space [27], by using an auto-encoder architecture [56], GAN-
like generator [70], or the combination of GANs and auto-encoders [75, 76]. Our work is independent of these different types of generators, and we focus primarily on how to train the generator to improve its use for recognition tasks by leveraging large amounts of auxiliary data and self-supervision.

**Few-Shot Learning and Meta-Learning.** Meta-learning, or the ability to learn to learn [64], is a powerful framework for tackling the problem of learning with limited data. Most of modern approaches fall into one of the categories between optimization and metric learning based methods. Optimization based methods learn how to do fast adaptation to novel tasks, by learning appropriate parameter updates [49] or a general initialization [21]. Adaptation could be done in the original feature space [21, 4, 6] or in an embedded space [52]. Prior work on few-shot domain adaptation [53, 31] learns how to balance cross-domain clustering that is domain invariant. Metric learning methods focus on learning a similarity metric [33]. Several distance functions have been explored, from the Euclidean distance [59, 2] and the cosine distance [13, 23, 19] to more complex parametric functions and metrics [68, 62, 37, 82], or using an additional task-specific metric [46]. Most methods often treat each category separately without considering the relations between them. Graph neural networks are thus introduced to leverage those relations [54, 32, 24]. To conduct meta-learning more effectively, recent approaches often first compute a set of features of the images using a trained feature extractor network. Given that high-dimensional features have better modeling capacity but are computationally expensive to work with, each meta-learning task is then formulated as a convex optimization problem and solved in its low-dimensional dual space [9, 34]. Our hallucinator component is generic and can be integrated into different meta-learning methods.

**Teacher-Student Networks.** Learning a model under the guidance of a teacher or mentor model has been widely used for model compression. Compressing one cumbersome or several models into a smaller model is a classic idea [18, 10] and has been popularized by the distillation formulation in [29]. Recent work focuses on advanced techniques to guide the distillation process [42, 78, 1] and its applications to practical problems, such as object detection [77, 73] and distributed machine learning [3]. In addition, knowledge distillation has been extended to address other tasks, including multi-task learning [63] and continual learning [38, 57]. To the best of our knowledge, our work is the first to introduce a mentor network for learning recognition task oriented generative models. Importantly, different from existing work that addresses models of different capacity, we consider models of the same capacity but trained on real or synthetic data.

**Contrastive Learning.** Powerful self-supervised representation learning approaches have recently been developed in image domain via manually specified pretext tasks. Examples include auto-encoding methods which leverage contrastive losses [47], channels [86], and colors [85] to recover the input under some corruption. Some pretext tasks form pseudo-labels by relative patch locations [17], image rotations [25], and jigsaw puzzles [44]. These pretext tasks are collected under the umbrella of the contrastive learning framework, which maintains the relative consistency between the representations of an image and its augmented views [45, 74, 80, 28, 12, 65, 26, 14, 81]. In our work, we treat hallucinated and real examples as different views of the data and use the contrastive loss to self-direct the hallucinated examples in the right class cluster or manifold.

### 3. Dual Mentor- and Self-Directed Hallucinator

**Few-Shot Learning Setting.** We are given a set of base categories \( C_{\text{base}} \) and a set of novel categories \( C_{\text{novel}} \), where \( C_{\text{base}} \cap C_{\text{novel}} = \emptyset \). We have a base dataset \( D_{\text{base}} \) with a large amount of annotated training examples per class and a novel dataset \( D_{\text{novel}} \) with only few annotated training examples per class. Few-shot learning aims to learn a good classification model \( h \) for \( C_{\text{novel}} \) based on the small dataset \( D_{\text{novel}} \). Recent work achieves this through a meta-learning procedure [68], which learns from a collection of sampled few-shot classification tasks on \( C_{\text{base}} \). Given a set of categories \( C \) and a set of data \( D \), an \( m \)-way \( k \)-shot task is composed of a subset \( C_{\text{sub}} \) of \( m \) categories from \( C \), a support (training) set \( S_{\text{supp}} \) of \( k \) examples from \( D \) for each class in \( C_{\text{sub}} \), and a query (test) set \( S_{\text{query}} \) of one or few examples from \( D \) for each class in \( C_{\text{sub}} \). Meta-learning is performed in two phases as follows.

During meta-training, a classifier learns from a collection of \( m \)-way \( k \)-shot tasks sampled from \( C_{\text{base}} \) and \( D_{\text{base}} \). While our work is agnostic to different classification models, here we take a simple cosine classifier [13] as an example – a variant of prototypical networks [59] which uses the cosine instead of the standard Euclidean distance function. In each iteration, we compute a prototype representation for each class in \( C_{\text{sub}} \). Each example is fed to an embedding function \( f_{\theta} \) with learnable parameters \( \theta \). The prototype of class \( c \) is the mean of the outputs through \( f_{\theta} \) of examples from \( C_{\text{sub}} \) in \( S_{\text{supp}} \). We then feed the examples in \( S_{\text{query}} \) to the classifier and update the parameters \( \theta \). During meta-testing, we use the same approach and build our previously meta-learned classifier with one unique \( m \)-way \( k \)-shot task, using \( C_{\text{novel}} \) instead of \( C_{\text{base}} \) and \( D_{\text{novel}} \) instead of \( D_{\text{base}} \). We evaluate the final classifier on unseen examples with labels from \( C_{\text{novel}} \).

**Meta-Learning with Hallucination.** Incorporating a generative model which produces additional examples for data augmentation has been shown to facilitate meta-learning [70, 22, 56]. While our approach does not rely on specific types of generative models, here we focus on the feature hallucinator in [70], due to its simplicity and state-of-the-art performance, which is implemented as a light-weight multi-layer perceptron (MLP) module. The hallucinator is a
function \( G(x, z; w) : \mathcal{R}^{d+d_{\text{noise}}} \rightarrow \mathcal{R}^{d} \) that produces examples in a pre-trained feature space of dimension \( d \), where \( x \) is the feature vector of a real example, \( z \) is a random noise vector of dimension \( d_{\text{noise}} \) sampled from a Gaussian distribution, and \( w \) is the parameters of \( G \). The hallucinated example \( G(x, z; w) \) is of the same category as \( x \).

Now the procedure of meta-learning integrated with the hallucinator \( G \) is illustrated in Figure 2. During each iteration of meta-training, the support set \( S_{\text{supp}} \) is first augmented by a generated set \( S_{\text{aug}}^{G} \). Specifically, for each class \( y \), we sample \( k_{\text{train}}^\text{gen} \) examples \((x, y)\) in \( S_{\text{supp}} \), sample associated random noise vectors \( z \), and then add \((x', y)\) to \( S_{\text{supp}}^{G} \), where \( x' = G(x, z; w) \). Our final support training set is \( S_{\text{supp}}^{\text{aug}} = S_{\text{supp}} \cup S_{\text{supp}}^{G} \). As long as \( G \) is differentiable with respect to the generated set \( S_{\text{supp}}^{G} \), the gradients of the final classification loss on \( S_{\text{query}} \) can be back-propagated into \( G \) to produce useful hallucinated examples. Through meta-training over a large amount of iterations, the hallucinator learns to capture shared modes of variation across different classes and can thus generalize to unseen classes. During meta-testing, we use the learned \( G \) to generate additional examples for recognizing categories in \( c_{\text{novel}} \).

**Hallucination with Extrinsic Guidance from Mentor.**

The end-to-end optimization of the classification loss enables the hallucinator to produce useful examples in the few-shot regime. However, since the classification loss is computed on small query sets, such supervision solely is insufficient to adjust the hallucinator to produce discriminative examples that most contribute to formulating classifier decision boundaries. Hence, the resulting classifier trained on the hallucinated examples could be still far away from the desired classifier that would be learned from a large set of real examples. This makes it critical to close the gap between these two classifiers. In fact, during meta-training, a large amount of annotated examples are already available for the base categories \( c_{\text{base}} \), which allows us to explicitly obtain the classifier trained on a large set of examples and use it to guide the learning of the hallucinator.

Formally, we treat the classifier trained on the augmented set of the hallucinated examples and the few support examples as a *student model*, and we treat the classifier trained on a large set of real base examples as a *mentor model*. Our goal then is to learn the hallucinator by minimizing the discrepancy between the student classifier and its mentor model. While a naïve approach would be to directly characterize the difference between their model parameters, it turns out to be challenging due to the high dimensionality of the parameter space. Inspired by the teacher-student network [29], we instead enforce the student to mimic the distribution of class probabilities predicted by the mentor network, which can be viewed as a way of regularization to improve the generalization performance of the student model [43].

As shown in Figure 2, meta-training the hallucinator \( G \) is conducted in the following way. We first sample a large set of examples \( S_{\text{large}} \), with \( k_{\text{large}} \) examples per class in \( c_{\text{base}} \), and train a mentor classifier using all the examples in \( S_{\text{large}} \). During each iteration of meta-training, we augment \( S_{\text{supp}} \) by generating new examples using the hallucinator \( G \). We train the student classifier on \( S_{\text{aug}}^{\text{supp}} \) through the knowledge distillation loss function in [29]:

\[
L_{\text{ex}}(s, m, y) = L_{\text{CE}}(\sigma(s), e_{y}) + \alpha \tau_{1}^{2} L_{\text{CE}}(\sigma\left(\frac{s}{\tau_{1}}\right), \sigma\left(\frac{m}{\tau_{1}}\right)),
\]

which consists of a standard cross-entropy loss (the first term) and an additional component that measures the difference between student and mentor outputs (the second
term. $s$ and $m$ are the logits produced by the student and the mentor, respectively, for a test example of label $y$ in $S_{\text{query}}$. $\sigma$ denotes the softmax function, $L_{\text{CE}}$ denotes the cross-entropy loss, $e_y$ is the one-hot encoding of $y$, and $\alpha$ is a trade-off hyper-parameter that balances the two terms. Note that $\tau_1 > 0$ is a critical learnable parameter called temperature, which smooths the probability distribution produced by the mentor and makes the corresponding decision boundary easier to learn for the student than the original one.

**Self-Directed Learning with Intrinsic Label Consistency.** While the hallucinated examples directed by the extrinsic mentor are useful, without other constraints they might spread over across class boundaries and thus be noisy. Inspired by supervised contrastive learning [30], we enforce intrinsic label consistency between hallucinated examples and real examples.

Formally, suppose we sample $N$ real examples per mini-batch and generate $M$ hallucinated examples, resulting in a batch $X$ of $M + N$ examples. Given an anchor example $x_i$, $P(i)$ is the set of indices of all positives in the batch distinct from $i$ and $A(i) = I_i \backslash \{i\}$. The supervised contrastive loss is defined as $L_{\text{in}} = \frac{1}{M+N} \sum_{i=1}^{M+N} L_i$ and

$$L_i = -\frac{1}{|P(i)|} \log \sum_{p \in P(i)} \frac{\exp(x_i \cdot x_p / \tau_2)}{\sum_{a \in A(i)} \exp(x_i \cdot x_a / \tau_2)},$$

(2)

where $\tau_2 > 0$ is a temperature parameter and $|P(i)|$ is its cardinality. This loss allows the real and hallucinated examples from the same classes to attract mutually, while they repel the other examples from different classes in the mini-batch.

Thus, our dual mentor- and self-directed hallucinators can be derived from Eqn. 1 and Eqn. 2 as

$$L = L_{\text{ex}} + \beta L_{\text{in}},$$

(3)

where $\beta$ is a trade-off hyper-parameter that balances the two terms. Minimizing Eqn. 3 over $S_{\text{query}}$ thus guides the hallucinator towards producing useful examples that help the student classifier recover the decision boundary from the mentor model.

**4. Progressive Guidance from Mentor Model**

Under the framework of meta-learning with extrinsic guidance, a straightforward way is to build the mentor model by using $k_{\text{large}}$ as large as possible (potentially the full set of $D_{\text{base}}$) and keep it fixed, and to train the hallucinator and student classifier using only few real examples. By doing so, however, we face the problem that the decision boundaries obtained by those two models could be very far from each other at the beginning of the training, making the learning of the hallucinator difficult. To address this issue, we perform the learning process in a progressive manner with varied number of real examples. We start with a mentor and a student which have access to a not too different number of real examples, and then progressively change the number of examples, so that the decision boundaries transform in a smooth manner. Concretely, this can be achieved in the following two dual directions.

**Progressive Guidance by Strengthening the Mentor.**

In this setting, both the student and the mentor start with a small number of real examples. However, the number of real examples for the mentor gradually increases over the training. The objective for the hallucinator then is to learn to generate additional examples so that its corresponding student can always match the performance of the mentor, whenever the mentor is re-trained with more samples and becomes stronger. More specifically, during meta-training, the support set $S_{\text{supp}}$ of each few-shot task is composed of very few examples per class, $k_{\text{train}}$, as in regular meta-training. At the beginning, we sample $S_{\text{large}}$, with $k_{\text{large}}$ being set to the value of $k_{\text{train}}$. We then progressively sample new real examples in the same amount for each class and add them into $S_{\text{large}}$. $k_{\text{large}}$ grows from $k_{\text{train}}$ to $k_{\text{max}}$ in a linear or logarithmic scale, where $k_{\text{max}}$ is the maximum available number of examples per class in $D_{\text{base}}$. We re-train the mentor model every time we add new examples.

**Progressive Guidance by Weakening the Student.**

In this setting, both the student and the mentor start with a large number of real examples. However, we gradually remove the real examples for the student over the training. The objective for the hallucinator then is to learn to generate the missing examples based on the remaining real examples. This allows the student to preserve or stabilize the original decision boundary formulated by the large set of examples (i.e., the mentor boundary), when the student has access to less real examples and becomes weaker. More specifically, during meta-training, the support set $S_{\text{supp}}$ of each “few-shot” task is composed of a large number of examples per class, unlike regular meta-training. This number of examples per class in $S_{\text{supp}}$, $k_{\text{train}}$, decreases in a linear or logarithmic scale, until it reaches a small value.

**5. Evaluation**

We now present experiments to evaluate our dual mentor- and self-directed (DMAS) hallucinator on few-shot classification, and study the effect of progressive guidance from extrinsic and intrinsic supervision. Since DMAS is agnos-
to the choice of classification models, we validate its
generalizability to different types of features and various
meta-learning models. In particular, we focus on simple
 cosine classifiers, which have been recently shown to achieve
very competitive few-shot performance [13].

Datasets. We evaluate on four widely-used datasets: (1)
 miniImageNet [68, 49], with 64, 16, and 20 classes for meta-
training, meta-validation, and meta-testing, respectively; (2)
tieredImageNet [51], with 20, 6, and 8 super-classes for
meta-training, meta-validation, and meta-testing, respectiv-
ely; (3) ImageNet1K [27, 70], with 193 base and 300
novel classes for cross-validation and 196 base and 311
novel classes for evaluation; (4) Caltech-UCSD Birds-200-
2011 (CUB) [69, 79], with 100, 50, and 50 classes for meta-
training, meta-validation, and meta-testing, respectively.

Implementation Details. For a fair comparison with pre-
vious work, we employ ResNet10 as our model backbone
for ImageNet1K [70] and ResNet12 as our model backbone
for the other three datasets [82]. As is commonly imple-
mented in the state-of-the-art work, we follow the feature
pre-training step [82]. We first train a convolutional network
based feature extractor on the base classes. Then we extract
and save these features to disk, and use these pre-computed
features as inputs for meta-learning. We follow the feature
hallucinator architecture in [70] and use a three layer MLP
with ReLU as the activation. The embedding function $f_{o}$
of our cosine classifier is a two layer MLP.

During progressive guidance by weakening the student,
we start training the mentor with $k_{large} = 256$, and we
increase the number to 1 in a logarithmic scale over 12,000
iterations. We initialize the learnable parameters including
the temperature $\tau_{1}$ to 7, the scale factor of the cosine distance
up to 75, and the temperature $\tau_{2}$ to 0.07. As the performance
is not sensitive to trade-off hyper-parameters $\alpha$ and $\beta$, we
empirically set them to 5 and 1, respectively. The number of
hallucinated examples is a hyper-parameter ranging from
2 – 10. The saturation point of hallucinated examples on
ImageNet is a hype-parameter ranging from

\[
\begin{array}{l}
\text{Method} & \text{Backbone} & \text{miniImageNet} & \text{tieredImageNet} \\
\hline
\text{Cosine Classifier [13]} & \text{ResNet12} & 55.43 \pm 0.81 & 77.18 \pm 0.61 \\
\text{TADAM [46]} & \text{ResNet12} & 58.50 \pm 0.30 & 76.70 \pm 0.30 \\
\text{ECM [50]} & \text{ResNet12} & 59.00 \pm 0.20 & 77.46 \pm 0.50 \\
\text{TPN [40]} & \text{ResNet12} & 59.46 \pm 0.20 & 76.55 \pm 0.46 \\
\text{PPA [48]} & \text{WRN-28-10} & 59.60 \pm 0.41 & 73.74 \pm 0.19 \\
\text{ProtoNet [59]} & \text{ResNet12} & 60.37 \pm 0.83 & 78.02 \pm 0.57 \\
\text{wD/AE-GNN [34]} & \text{WRN-28-10} & 61.07 \pm 0.15 & 76.75 \pm 0.11 \\
\text{MTL [61]} & \text{ResNet12} & 61.20 \pm 0.80 & 75.30 \pm 0.80 \\
\text{DC [39]} & \text{ResNet12} & 61.76 \pm 0.08 & 77.39 \pm 0.12 \\
\text{MetaOptNet [34]} & \text{ResNet12} & 62.64 \pm 0.19 & 79.77 \pm 0.19 \\
\text{FEAT [79]} & \text{ResNet24} & 62.96 \pm 0.20 & 78.49 \pm 0.15 \\
\text{MatchingNet [68]} & \text{ResNet12} & 63.08 \pm 0.80 & 75.99 \pm 0.60 \\
\text{CTM [35]} & \text{ResNet18} & 64.12 \pm 0.82 & 80.51 \pm 0.13 \\
\text{RFS [66]} & \text{ResNet12} & 64.82 \pm 0.60 & 82.14 \pm 0.43 \\
\text{DeepEMD [82]} & \text{ResNet12} & 65.91 \pm 0.82 & 82.41 \pm 0.56 \\
\hline
\text{DMAS (Ours)} & \text{ResNet12} & 67.42 \pm 0.28 & 83.74 \pm 0.20 \\
\end{array}
\]
Table 2: Ablation studies (top-5 accuracy) on ImageNet1K 311-way classification: (1) different pre-trained feature spaces for hallucination – ‘standard’ (the feature backbone is a ResNet10 pre-trained using a standard cross-entropy linear classifier on base classes) vs. ‘cosine’ (the ResNet10 feature backbone is pre-trained using a cosine classifier); (2) different types of classifiers – prototypical net vs. cosine classifier; (3) impact of different sources of supervision and progressive training. w/ aug: with standard data augmentation. Gen: with a plain hallucinator [70] trained using the classification loss on the query set solely. DMAS w/ ex: DMAS trained only under the guidance of the mentor without progressive training. DMAS w/ ex↑: progressive guidance through strengthening the mentor. DMAS w/ ex↓: progressive guidance through weakening the student. DMAS w/ in: DMAS trained only in a self-directed way through contrastive learning. DMAS (full): trained under both (progressively) extrinsic and intrinsic supervision.

Table 3: Cross-domain evaluation (miniImageNet → CUB). Our model outperforms other baseline methods by large margins, showing the generalization of our learned hallucinator.

Table 4: Ablation study on the generalizability of our approach and additional comparisons with state of the art on miniImageNet. Our DMAS hallucinator is general and can work with different types of classification models and different backbone models to consistently improve their performance. In addition, DMAS consistently outperforms the plain hallucinator [70].
Figure 4: Visualization with t-SNE of the evolution of the decision boundary for two novel classes, when meta-training our DMAS hallucinator through progressive guidance by weakening the student. Real examples (small dots) are progressively removed, and hallucinated examples (triangles) are generated in a way that helps maintain the student decision boundary (black solid line) as close as possible to the desired decision boundary that would be formulated by a large set of real examples (red dashed line). We observe that PCA visualization has a similar phenomenon. **Best viewed in color with zoom.**

Figure 5: Visualization of nearest neighbor real images of hallucinated examples for four novel classes. For each class, the single black framed image comes from the original dataset and is used as a seed for generating new examples. Color framed images correspond to the nearest neighbor real images of the hallucinated examples in the feature space. **Best viewed in color with zoom.**

Figure 6: Visualization of classification results of two novel classes (Top row: malamute; bottom row: mixing bowl) and comparison between our DMAS hallucinator and the plain hallucinator [70]. The left block shows images correctly classified by both approaches. The middle block shows images that are misclassified by [70] as other classes (with predicted class names overlaid on the images), but correctly classified by our approach. The right block shows images from other classes that are misclassified by [70] as the target class, but correctly classified by our approach. In these examples, our classifier is able to recognize objects with different poses and view points, whereas [70] fails to distinguish between similar classes.

Comparisons with the plain hallucinator. Table 4 also shows that DMAS consistently outperforms the plain hallucinator [70] for different types of models (Table 2 has already shown this for ProtoNet and cosine classifier). More importantly, ‘DeepEMD + [70]’ is worse than the plain DeepEMD; a similar phenomenon is observed with S2M2 in the 1-shot case. These results suggest that, while DMAS is general, [70] is not a general module for different few-shot models. For more sophisticated models (S2M2 and DeepEMD), solely using the classification loss as in [70] is insufficient to adjust the hallucinator to produce effective samples. This further verifies the importance of extrinsic and intrinsic supervision.

Visualizations. To further understand how our model helps learning a classifier and refining the hallucinator, we conduct visualizations on ImageNet1K. We first visualize in Figure 4 the evolution of the decision boundary for two novel classes during progressive guidance by weakening the student using t-SNE [67]. We then visualize in Figure 5 the hallucinated examples in the pixel space, using their nearest neighbor real images in the feature space. Finally in Figure 6, we compare our approach with the state-of-the-art meta-learned hallucinator [70] and show that ours is able to recognize a large range of visual variations.

6. Conclusion

We present an approach to few-shot classification that uses a dual mentor- and self-directed hallucinator to generate additional examples. This is achieved by exploiting two important natural supervision signals that facilitate data hallucination in a way that most improves the classification performance, and is trained end-to-end through meta-learning. Our hallucinator can be inserted as a plug-and-play module into different classification models. The extensive experiments demonstrate our state-of-the-art performance on the widely-benchmarked few-shot datasets in various scenarios.

Acknowledgement: This work was supported in part by ONR MURI N00014-16-1-2007, AFRL Grant FA23861714660, and NSF Grant 2106825. We also thank AWS Cloud Credits for Research program.
References

[38] Zhizhong Li and Derek Hoiem. Learning without forgetting. In TPAMI, 40(12):2935–2947, 2017. 3


[78] Zheng Xu, Yen-Chang Hsu, and Jiawei Huang. Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks. In ICLR, 2018.


