Data Augmentation for Meta-Learning

Renkun Ni [1] Micah Goldblum [1] Amr Sharaf [2] Kezhi Kong [1] Tom Goldstein [1]

Abstract

Conventional image classifiers are trained by randomly sampling mini-batches of images. To achieve state-of-the-art performance, practitioners use sophisticated data augmentation schemes to expand the amount of training data available for sampling. In contrast, meta-learning algorithms sample support data, query data, and tasks on each training step. In this complex sampling scenario, data augmentation can be used not only to expand the number of images available per class, but also to generate entirely new classes/tasks. We systematically dissect the meta-learning pipeline and investigate the distinct ways in which data augmentation can be integrated at both the image and class levels. Our proposed meta-specific data augmentation significantly improves the performance of meta-learners on few-shot classification benchmarks.

1. Introduction

Data augmentation has become an essential part of the training pipeline for image classifiers and similar systems, as it offers a simple and efficient way to significantly improve performance (Cubuk et al., 2018; Zhang et al., 2017). In contrast, little work exists on data augmentation for meta-learning. Existing frameworks for few-shot image classification use only horizontal flips, random crops, and color jitter to augment images in a way that parallels augmentation for conventional training (Bertinetto et al., 2018; Lee et al., 2019). Meanwhile, meta-learning methods have received increasing attention as they have reached the cutting edge of few-shot performance. While new meta-learning algorithms emerge at a rapid rate, we show that, like image classifiers, meta-learners can achieve significant performance boosts through carefully chosen data augmentation strategies that are injected into various stages of the meta-learning pipeline.

[1]Department of Computer Science, University of Maryland, College Park [2]Microsoft. Correspondence to: Renkun Ni <rn9zm@cs.umd.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Meta-learning frameworks use data for multiple purposes during each gradient update, which creates the possibility for a diverse range of data augmentations that are not possible within the standard training pipeline. At the same time, it is still unclear how different categories of data within the training pipeline impact meta-learning performance. We explore these possibilities and discover combinations of augmentation types that improve performance over existing methods. Our contributions can be summarized as follows:

- First, we break down the meta-learning pipeline and find that each component contributes differently to meta-learning performance: meta-learners are very sensitive to the amount of query data and number of tasks and less sensitive to the amount of support data.
- Based on these findings, we uncover four modes of augmentations for meta-learning that differ in where in the training pipeline they are applied: support augmentation, query augmentation, task augmentation, and shot augmentation.
- We test these four modes using a pool of image augmentations, and we confirm that query augmentation is critical, while support augmentation often does not provide performance benefits and may even degrade accuracy in some cases.
- Finally, we combine augmentations and implement a MaxUp strategy, which we call Meta-MaxUp, to maximize performance. We achieve significant performance boosts for popular meta-learners on few-shot benchmarks such as mini-ImageNet, CIFAR-FS and Meta-Dataset.

2. Background and Related Work

2.1. The Meta-Learning Framework

Meta-learning algorithms aim to learn a network that can easily adapt to new tasks with limited data and generalize to unseen examples. In order to achieve this, they simulate the adaptation and evaluation procedure during meta-training. To simulate an N-way classification task, $\mathcal{T}_i$, we sample *support* data $\mathcal{T}_i^s$ and *query* data $\mathcal{T}_i^q$, so that $\mathcal{T}_i = \{\mathcal{T}_i^s, \mathcal{T}_i^q\}$. As we will detail in the following paragraph, support will be used to simulate few-shot training data, while query will be used to simulate unseen testing data. Note that *shot* denotes the number of training samples per class available for fine-tuning on a given task during the testing phase.

Adopting common terminology from the literature, the archetypal meta-learning algorithm contains an *inner loop* and an *outer loop* in each parameter update of the training procedure. In the inner loop, a model is first fine-tuned or adapted on support data $\mathcal{T}_i^s$. Then, in the outer loop, the updated model is evaluated on query data $\mathcal{T}_i^q$, and minimizes loss on the query data with respect to the model's parameters before fine-tuning. This loss minimization step may require computing the gradient through the fine-tuning procedure. Existing meta-learning algorithms apply various methods for fine-tuning on support data during the inner loop. Some algorithms, such as MAML and Reptile (Finn et al., 2017; Nichol et al., 2018), update all the parameters in the network using gradient descent during fine-tuning on support data. Other algorithms, such as MetaOptNet and R2-D2 (Lee et al., 2019; Bertinetto et al., 2018), only update the parameters from the linear classifier layer during the fine-tuning while keeping the feature extraction layers frozen. These methods benefit from the simplicity and the convexity of the inner loop optimization problem. Similarly, metric learning approaches, such as (Snell et al., 2017; Kye et al., 2020), freeze the feature extraction layers as well, and create class centroids from the support data during the inner loop. These method have low cost training iterations, and can be applied on deeper architectures to achieve better performance. In this work, we mainly focus on the latter algorithms due to their stronger performance. Further details of the algorithms used in our experiments can be found in Section 4.1.

2.2. Preventing Overfitting in Meta-Learning

Meta-learners are known to be particularly vulnerable to overfitting (Rajendran et al., 2020). One work, MetaMix, proposes averaging support and query features to prevent the model from memorizing the query data and ignoring support (Yao et al., 2020). Recently, another work adds random noise to the label space to make the model rely on support data (Rajendran et al., 2020). In the context of few-shot classification, random shuffling labels within tasks alleviates this kind of overfitting and is commonplace in meta-learning algorithms (Yin et al., 2019; Rajendran et al., 2020). However, as shown in Figure 1, overfitting to training tasks remains a problem. One recent work has developed a data augmentation method to overcome this problem (Liu et al., 2020). This method simply rotates all images in a class by a large degree and considers this new rotated class distinct from its parent class. This effectively increases the number of possible few-shot tasks that can be sampled during training.

A different line of work instead applies regularizers to prevent overfitting and improve few-shot classification (Yin et al., 2019; Goldblum et al., 2020). Yet additional work has developed methods for labeling and augmenting unlabeled data (Antoniou & Storkey, 2019; Chen et al., 2019b), generative models for deforming images in one-shot metric learning (Chen et al., 2019c), and feature space data augmentation for adapting language models to new unseen intents (Kumar et al., 2019).

2.3. Few-shot Benchmarks

In this paper, we perform our experiments on the mini-ImageNet and CIFAR-FS datasets as well as the Meta-Dataset benchmark (Vinyals et al., 2016; Bertinetto et al., 2018; Triantafillou et al., 2019). Mini-ImageNet is a few-shot learning dataset derived from the ImageNet classification dataset (Deng et al., 2009), and CIFAR-FS is derived from CIFAR-100 (Krizhevsky et al., 2009). Each of these datasets contains 64 training classes, 16 validation classes, and 20 classes for testing. In each class, there are 600 images, and both Mini-ImageNet and CIFAR-FS have 60000 images in total. Meta-Dataset is a large-scale diverse benchmark consisting of 10 different image classification subdatasets with distinct data distributions. This diversity allows us to measure cross-domain generalization.

3. The Anatomy of Data Augmentation for Meta-Learning

3.1. Where Does Dataset Diversity Matter Most? In the Support, Query or Tasks?

Since data augmentation techniques aim to increase the amount of training samples, learning algorithms that are sensitive to the amount of training data may benefit more from these techniques. In this section, before we introduce data augmentations, we investigate how sensitive meta-learning algorithms are to the amount of support data, query data, and tasks. Typically, support and query data are sampled from the same pool (the entire training set).

To examine the impact of dataset diversity on various stages of meta-learning, we perform an ablation where we limit the diversity of each stage. We first reduce the pool of support data to a fixed subset of only five independent samples per class while sampling query data from the entire training set. That is, whenever a support image is sample from class c, it is only sampled from the five-image subset associated with that class instead of from all training data in that class. Interestingly, we find that test accuracy remains almost the same as baseline performance (see Table 1). In fact, if we replace those five support images per class with fixed random noise images, we still only observe a small degradation in performance. We then instead shrink the pool of query data (but not support), and we see a much larger decrease in test accuracy. These experiments suggest that meta-learning is fairly insensitive to the amount and quality of support but not query data. This observation agrees with our following finding that augmenting query data is far more beneficial than augmenting support.

Since we also consider task-level augmentation, we now examine how sensitive meta-learning is to a decrease in task diversity. As CIFAR-FS contains 64 training classes, there are $\binom{64}{5} = 7624512$ 5-way classification problems that can be sampled during each iteration of meta-learning. We reduce the number of tasks by randomly batching classes into just 13 distinct 5-way classification tasks before training, and we only train on these 13 tasks. We do this in such a way that all classes, and therefore training data, are used during training. We observe that this process noticeably degrades test accuracy, and we conclude that there may be room to improve performance by augmenting the number of tasks (see Table 1). To verify that this impact of dataset diversity generalizes, we run additional experiments on Mini-ImageNet and with other backbones. The results are shown in Appendix A, and these experiments support the aforementioned findings as well.

Table 1. Few-shot classification accuracy (%) using R2-D2 and a ResNet-12 backbone for various data size manipulations on CIFAR-FS. "Support", "Query" and "Task" columns denote the number of samples per class for support and query data and the number of total tasks available for sampling. The first row contains baseline performance. Confidence intervals have radius equal to one standard error.

Support	Query	Task	1-shot	5-shot
600	600	full	71.73 ± 0.37	84.39 ± 0.25
5	600	full	70.97 ± 0.36	84.51 ± 0.24
5 (random)	600	full	58.15 ± 0.36	76.26 ± 0.27
600	5	full	60.25 ± 0.37	77.05 ± 0.28
600	600	13	68.24 ± 0.38	81.77 ± 0.26

3.2. Data Augmentation Modes

Motivated by the observation that meta-learning is more sensitive to the amount of query data and tasks than support, we delineate four modes of data augmentation for meta-learning which may be employed individually or combined.

Support augmentation: Data augmentation may be applied to support data in the inner loop of fine-tuning. This strategy enlarges the pool of fine-tuning data.

Query augmentation: Data augmentation alternatively may be applied to query data. This strategy enlarges the pool of evaluation data to be sampled during training.

Task augmentation: We can increase the number of possible tasks by uniformly augmenting whole classes to add new classes with which to train. For example, a vertical flip applied to all car images yields a new upside-down car class which may be sampled during training.

Shot augmentation: At test time, we can artificially amplify the shot by adding additional augmented copies of each image. Shot augmentation can also be used during training by adding copies of each support image via augmentation. Shot augmentation during training may be needed to prepare a network for the use of test-time shot augmentation.

Existing meta-learning algorithms for few-shot image classification typically apply standard augmentations (horizontal flips, random crops, and color jitter) on all images that come from the data loader without considering the purpose of each image. As a result, the same augmentation occurs on both support and query images (Gidaris & Komodakis, 2018; Qiao et al., 2018). In Section 4, we test the four modes of data augmentation enumerated above in isolation across a large array of specific augmentations. We find that query augmentation is far more critical than support augmentation for increasing performance. In fact, support augmentation often hurts performance. Additionally, we find that task augmentation, when combined with query augmentation, can offer further boosts in performance when compared with existing frameworks.

3.3. Data Augmentation Techniques

For each of the data augmentation modes described above, we try a variety of specific data augmentation techniques. Some techniques are only applicable to support, query, and shot modes or solely to the task mode. We use an array of standard augmentation techniques as well as CutMix (Yun et al., 2019), MixUp (Zhang et al., 2017), and Self-Mix (Seo et al., 2020). In the context of the task augmentation mode, we apply these the same way to every image in a class in order to augment the number of classes. For example, we use MixUp to create a half-dog-half-truck class where every image is the average of a dog image and a truck image. We also try combining multiple classes into one class as a task augmentation mode.

In general, techniques that greatly change the image distribution (i.e. a vertical flip, which does not naturally appear in the dataset) are better suited for task augmentations while techniques that preserve the image distribution (e.g., random crops, which produce images that are presumably within the support of the image distribution) are typically better suited for the support, query, and shot augmentation modes. The baseline models we compare to use horizontal flip, random crop, and color jitter augmentation techniques at both the support and query levels since this combination is prevalent in the literature. More details on our pool of augmentation techniques can be found in Appendix B.

3.4. Meta-MaxUp Augmentation for Meta-Learning

Recent work proposes MaxUp augmentation to alleviate overfitting during the training of classifiers (Gong et al., 2020). This strategy applies many augmentations to each image and chooses the augmented image which yields the highest loss. MaxUp is conceptually similar to adversarial training (Madry et al., 2019). Like adversarial training, MaxUp involves solving a saddlepoint problem in which loss is minimized with respect to parameters while being maximized with respect to the input. In the standard image classification setting, MaxUp, together with CutMix, improves generalization and achieves state-of-the-art performance on ImageNet. Here, we extend MaxUp to the setting of meta-learning. Before training, we select a pool, $\mathcal{S}$, of data augmentations from the four modes as well as their combinations. For example, $\mathcal{S}$ may contain horizontal flip shot augmentation, query CutMix, and the combination of both. During each iteration of training, we first sample a batch of tasks, each containing support and query data, as is typical in the meta-learning framework. For each element in the batch, we randomly select m augmentations from the set $\mathcal{S}$, and we apply these to the task, generating m augmented tasks with augmented support and query data. Then, for each element of the batch of tasks originally sampled, we choose the augmented task that maximizes loss, and we perform a parameter update step to minimize training loss. Formally, we solve the minimax optimization problem,

$$\min_{\theta} \mathbb{E}_{\mathcal{T}} \Big[\max_{M \in \mathcal{S}} \mathcal{L}(F_{\theta'}, M(\mathcal{T}^q)) \Big],$$

where $\theta' = \mathcal{A}(\theta, M(\mathcal{T}^s))$, $\mathcal{A}$ denotes fine-tuning, F is the base model with parameters θ, $\mathcal{L}$ is the loss function used in the outer loop of training, and $\mathcal{T}$ is a task with support and query data $\mathcal{T}^s$ and $\mathcal{T}^q$, respectively. Algorithm 1 contains a more thorough description of this pipeline in practice (adapted from the standard meta-learning algorithm in Goldblum et al. (2019)).

Algorithm 1 Meta-MaxUp

```
Require: Base model $F_\theta$, fine-tuning algorithm $\mathcal{A}$, learning rate $\gamma$, set of augmentations $\mathcal{S}$, and distribution over tasks $p(\mathcal{T})$.
Initialize $\theta$, the weights of $F$;
while not done do
    Sample batch of tasks, $\{\mathcal{T}_i\}_{i=1}^n$, where $\mathcal{T}_i \sim p(\mathcal{T})$ and $\mathcal{T}_i = (\mathcal{T}_i^s, \mathcal{T}_i^q)$.
    for $i = 1, ..., n$ do
        Sample $m$ augmentations, $\{M_j\}_{j=1}^m$, from $\mathcal{S}$.
        Compute $k = \text{argmax}_j \mathcal{L}(F_{\theta_j}, M_j(\mathcal{T}_i^q))$, where $\theta_j = \mathcal{A}(\theta, M_j(\mathcal{T}_i^s))$.
        Compute gradient $g_i = \nabla_\theta \mathcal{L}(F_{\theta_k}, M_k(\mathcal{T}_i^q))$.
    end for
    Update base model parameters: $\theta \leftarrow \theta - \frac{\gamma}{n} \sum_i g_i$.
end while
```

4. Experiments

In this section, we empirically demonstrate the following:

1. Augmentations applied in the four distinct modes behave differently. In particular, query and task augmentation are far more important than support augmentation. (Section 4.2)

2. Meta-specific data augmentation strategies can improve performance over the generic strategies commonly used for meta-learning. (Section 4.3)

3. We further boost performance by combining augmentations with Meta-MaxUp. (Section 4.4)

4. Our proposed augmentation Meta-MaxUp greatly improves performance on cross-domain benchmarks as well. (Section 4.7)

4.1. Experimental Setup

We conduct experiments on four meta-learning algorithms: ProtoNet (Snell et al., 2017), R2-D2 (Bertinetto et al., 2018), MetaOptNet (Lee et al., 2019), and MCT (Kye et al., 2020). ProtoNet is a metric-learning method that uses a prototype learning head, which classifies samples by extracting a feature vector and then performing a nearest-neighbor search for the closest class prototype. R2-D2 and MetaOptNet instead use differentiable solvers with a ridge regression and SVM head, respectively. These methods extract feature vectors and then apply a standard linear classifer to assign class labels. MCT improves upon ProtoNet by meta-learning confidence scores. We experiment with all of these different classifier head options, all using the ResNet-12 backbone proposed by Oreshkin et al. (2018) as well as the four-layer convolutional architectures proposed by Snell et al. (2017) and Bertinetto et al. (2018).

We perform our experiments on the aforementioned benchmark datasets, mini-ImageNet, CIFAR-FS, and Meta-Dataset. A description of training hyperparameters and computational complexity can be found in Appendix C. We report confidence intervals with a radius of one standard error.

Few-shot learning may be performed in either the inductive or transductive setting. Inductive learning is a standard method in which each test image is evaluated separately and independently. In contrast, transduction is a mode of inference in which the few-shot learner has access to all unlabeled testing data at once and therefore has the ability to perform semi-supervised learning by training on the unlabelled data. For fair comparison, we only compare inductive methods to other inductive methods. A PyTorch implementation of our data augmentation methods for meta-learning can be found at: `https://github.com/RenkunNi/MetaAug`

4.2. An Empirical Comparison of Augmentation Modes

We empirically evaluate the performance of all four different augmentation modes identified in Section 3.2 on the CIFAR-FS dataset using an R2-D2 base-learner paired with both a 4-layer convolutional network backbone (as used in the original work (Bertinetto et al., 2018)) and a ResNet-12 backbone. We report the results of the most effective augmentations for each mode on the ResNet-12 backbone in Table 2. Appendix D contains an extensive table with various augmentations and both backbones.

Table 2 demonstrates that each mode of augmentation individually can improve performance. Augmentation applied to query data is consistently more effective than the other augmentation modes. In particular, simply applying CutMix to query samples improves accuracy by as much as 3% on both backbones. In contrast, most augmentations on support data actually damage performance. The overarching conclusion of these experiments is that the four modes of data augmentation for meta-learning behave differently. Existing meta-learning methods, which apply the same augmentations to query and support data without using task and shot augmentation, may be achieving suboptimal performance.

Table 2. Few-shot classification accuracy (%) using R2-D2 and a ResNet-12 backbone on the CIFAR-FS dataset with the most effective data augmentations for each mode shown. Confidence intervals have radii equal to one standard error. Best performance in each category is bolded. Query CutMix is consistently the most effective single augmentation for meta-learning.

Method	Mode	1-shot	5-shot
Baseline	-	71.95 ± 0.37	84.56 ± 0.25
CutMix	Support	72.79 ± 0.37	84.70 ± 0.25
Self-Mix	Support	71.96 ± 0.36	84.84 ± 0.25
CutMix	Query	$\mathbf{75.97 \pm 0.34}$	$\mathbf{87.28 \pm 0.23}$
Self-Mix	Query	73.59 ± 0.35	86.14 ± 0.24
Large Rotation	Task	73.79 ± 0.36	85.81 ± 0.24
MixUp	Task	72.05 ± 0.37	85.27 ± 0.25
Random Crop	Shot	70.56 ± 0.37	83.87 ± 0.25
Horizontal Flip	Shot	73.25 ± 0.36	85.06 ± 0.25

Table 3. Few-shot classification accuracy (%) using R2-D2 and a ResNet-12 backbone on the CIFAR-FS dataset with combinations of augmentations and query CutMix. "S","Q","T" denote "Support", "Query", and "Task" modes, respectively. While adding augmentations can help, it can also hurt, so additional augmentations must be chosen carefully.

Mode	1-shot	5-shot
CutMix	75.97 ± 0.34	87.28 ± 0.23
+ CutMix (S)	75.00 ± 0.37	85.37 ± 0.25
+ Random Erase (S)	75.84 ± 0.34	87.19 ± 0.24
+ Random Erase (Q)	75.08 ± 0.35	87.14 ± 0.23
+ Self-Mix (S)	$\mathbf{76.27 \pm 0.34}$	87.52 ± 0.24
+ Self-Mix (Q)	76.04 ± 0.34	87.45 ± 0.24
+ MixUp (T)	75.97 ± 0.34	86.66 ± 0.24
+ Rotation (T)	75.74 ± 0.34	$\mathbf{87.68 \pm 0.24}$
+ Horizontal Flip (Shot)	76.23 ± 0.34	87.36 ± 0.24

4.3. Combining Augmentations

After studying each mode of data augmentation individually, we combine augmentations in order to find out how augmentations interact with each other. We build on top of query CutMix since this augmentation was the most effective in the previous section. We combine query CutMix with other effective augmentations from Table 2, and we conduct experiments on the same backbones and dataset. Results on the ResNet-12 backbone are reported in Table 3, and a full table with additional results can be find in Appendix E. Interestingly, when we use CutMix on both support and query images, we observe worse performance than simply using CutMix on query data alone. Again, this demonstrates that meta-learning demands a careful and meta-specific data augmentation strategy. In order to further boost performance, we will need an intelligent method for combining various augmentations. We propose Meta-MaxUp as this method.

4.4. Meta-MaxUp Further Improves Performance

In this section, we evaluate our proposed Meta-MaxUp strategy in the same experimental setting as above for various values of m and different data augmentation pool sizes. Table 4 contains the results, and a detailed description of the augmentation pools as well as the full results can be found in Appendix F. Rows beginning with "CutMix" denote experiments in which the pool of augmentations simply includes many CutMix samples. "Single" denotes experiments in which each augmentation in $\mathcal{S}$ is of a single type, while "Medium" and "Large" denote experiments in which each element of $\mathcal{S}$ is a combination of augmentations, for example CutMix+rotation. Combinations greatly expand the number of augmentations in the pool. Rows with $m = 1$ denote experiments where we do not maximize loss in the inner loop and thus simply apply randomly sampled data

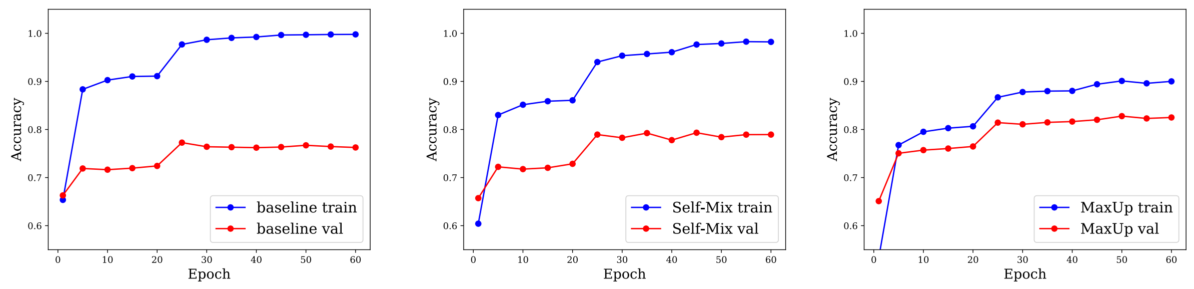

Figure 1. Training and validation accuracy for R2-D2 meta-learner with ResNet-12 backbone on the CIFAR-FS dataset. (Left) Baseline model (Middle) query Self-Mix (Right) Meta-MaxUp. Better data augmentation strategies, such as MaxUp, narrow the generalization gap and prevent overfitting.

augmentation for each task. As we increase m and include a large number of augmentations in the pool, we observe performance boosts as high as 4% over the baseline, which uses horizontal flip, random crop, and color jitter data augmentations from the original work corresponding to the R2-D2 meta-learner (Bertinetto et al., 2018).

Table 4. Few-shot classification accuracy ($\%$) using R2-D2 and a ResNet-12 backbone on the CIFAR-FS dataset for Meta-MaxUp over different sizes of augmentation pools and numbers of samples. As m and the pool size increase, so does performance. Meta-MaxUp is able to pick effective augmentations from a large pool.

Pool	m	1-shot	5-shot
Baseline	-	71.95 ± 0.37	84.56 ± 0.25
CutMix	1	75.97 ± 0.34	87.28 ± 0.23
Single	1	75.71 ± 0.35	87.44 ± 0.43
Medium	1	75.60 ± 0.34	87.35 ± 0.23
Large	1	75.44 ± 0.34	87.47 ± 0.23
CutMix	2	74.93 ± 0.36	87.14 ± 0.24
Single	2	75.81 ± 0.34	87.33 ± 0.23
Medium	2	76.49 ± 0.33	88.20 ± 0.22
Large	2	76.59 ± 0.34	88.11 ± 0.23
CutMix	4	75.08 ± 0.23	87.60 ± 0.24
Single	4	76.82 ± 0.24	88.14 ± 0.23
Medium	4	76.30 ± 0.24	88.29 ± 0.22
Large	4	**76.99 ± 0.24**	**88.35 ± 0.22**

We explore the training benefits of these meta-specific training schemes by examining saturation during training. To this end, we plot the training and validation accuracy over time for R2-D2 meta-learners with ResNet-12 backbones using baseline augmentations, query Self-Mix, and Meta-MaxUp with a medium sized pool and $m = 4$. See Figure 1 for training and validation accuracy curves. With only baseline augmentations, validation accuracy stops increasing immediately after the first learning rate decay. This suggests that baseline augmentations do not prevent overfitting during meta-training. In contrast, we observe that models trained with Meta-MaxUp do not quickly overfit and continue improving validation performance for a greater number of epochs. Meta-MaxUp visibly reduces the generalization gap.

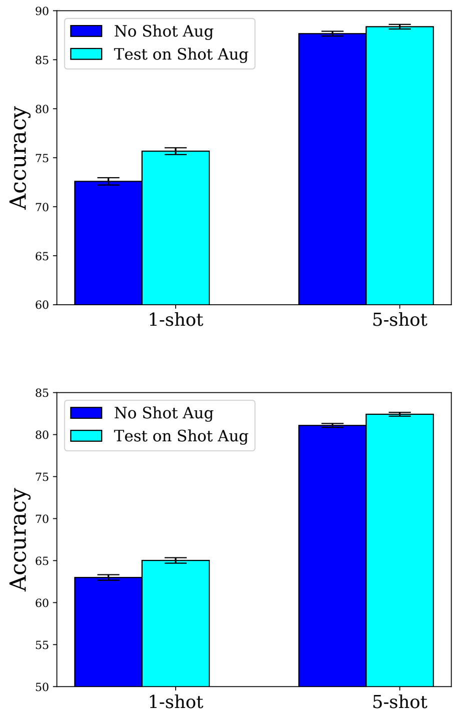

Figure 2. Performance with shot augmentation using MetaOptNet trained with the proposed Meta-MaxUp. (Top) 1-shot and 5-shot on CIFAR-FS (Bottom) 1-shot and 5-shot on mini-ImageNet.

4.5. Shot Augmentation for Pre-Trained Models

In the typical meta-learning framework, data augmentations are used during meta-training but not during test time. On the other hand, in some transfer learning work, data augmentations, such as horizontal flips, random crops, and color

Table 5. Few-shot classification accuracy (%) on CIFAR-FS and mini-ImageNet. "+ DA" denotes training with CutMix (Q) + Rotation (T), and "+ MM" denotes training with Meta-MaxUp. "CNN-4" denotes a 4-layer convolutional network with 96, 192, 384, and 512 filters in each layer (Bertinetto et al., 2018). "64-64-64-64" denotes the 4-layer CNN backbone from Snell et al. (2017).

		CIFAR-FS		mini-ImageNet	
Method	Backbone	1-shot	5-shot	1-shot	5-shot
R2-D2	CNN-4	67.56 ± 0.35	82.39 ± 0.26	56.15 ± 0.31	72.46 ± 0.26
+ DA	CNN-4	70.54 ± 0.33	84.69 ± 0.24	57.60 ± 0.32	74.69 ± 0.25
+ MM	CNN-4	**71.10 ± 0.34**	**85.50 ± 0.24**	**58.18 ± 0.32**	**75.35 ± 0.25**
R2-D2	ResNet-12	71.95 ± 0.37	84.56 ± 0.25	60.46 ± 0.32	76.88 ± 0.24
+ DA	ResNet-12	76.17 ± 0.34	87.74 ± 0.24	**65.54 ± 0.32**	81.52 ± 0.23
+ MM	ResNet-12	**76.65 ± 0.33**	**88.57 ± 0.24**	65.15 ± 0.32	**81.76 ± 0.24**
ProtoNet	64-64-64-64	60.91 ± 0.35	79.73 ± 0.27	47.97 ± 0.32	70.13 ± 0.27
+ DA	64-64-64-64	62.21 ± 0.36	80.70 ± 0.27	**50.38 ± 0.32**	**71.44 ± 0.26**
+ MM	64-64-64-64	**63.01 ± 0.36**	**80.85 ± 0.25**	50.06 ± 0.32	71.13 ± 0.26
ProtoNet	ResNet-12	70.21 ± 0.36	84.26 ± 0.25	57.34 ± 0.34	75.81 ± 0.25
+ DA	ResNet-12	74.30 ± 0.36	86.24 ± 0.24	60.82 ± 0.34	78.23 ± 0.25
+ MM	ResNet-12	**76.05 ± 0.34**	**87.84 ± 0.23**	**62.81 ± 0.34**	**79.38 ± 0.24**
MetaOptNet	ResNet-12	70.99 ± 0.37	84.00 ± 0.25	60.01 ± 0.32	77.42 ± 0.23
+ DA	ResNet-12	74.56 ± 0.34	87.61 ± 0.23	64.94 ± 0.33	82.10 ± 0.23
+ MM	ResNet-12	**75.67 ± 0.34**	**88.37 ± 0.23**	**65.02 ± 0.32**	**82.42 ± 0.23**
MCT	ResNet-12	75.80 ± 0.33	89.10 ± 0.42	64.84 ± 0.33	81.45 ± 0.23
+ MM	ResNet-12	**76.00 ± 0.33**	**89.54 ± 0.33**	**66.37 ± 0.32**	**83.11 ± 0.22**

jitter, are used during fine-tuning at test time (Chen et al., 2019a). These techniques enable the network to see more data samples during few-shot testing, leading to enhanced performance.

We propose shot augmentation (see Section 3) to enlarge the number of few-shot samples during testing, and we also propose a variant in which we additionally train using the same augmentations on support data in order to prepare the meta-learner for this test time scenario. Figure 2 shows the effect of shot augmentation (using only horizontal flips) on performance for MetaOptNet with ResNet-12 backbone trained with Meta-MaxUp. Shot augmentation consistently improves results across datasets, especially on 1-shot classification ($\sim 2\%$). To be clear, in this figure, we are not using shot augmentation during the training stage. Rather, we are using conventional low-shot training, and then deploying our models with shot augmentation at test time. These post-training performance gains can be achieved by directly applying shot augmentation to pre-trained/existing models during testing. For additional experiments, see Appendix G.

4.6. Improving Existing Meta-Learners with Better Data Augmentation

In this section, we improve the performance of four different popular meta-learning methods including ProtoNet (Snell et al., 2017), R2-D2 (Bertinetto et al., 2018), MetaOptNet (Lee et al., 2019), and MCT (Kye et al., 2020). We compare their baseline performance to query CutMix with task-level rotation as well as Meta-MaxUp data augmentation strategies on both the CIFAR-FS and mini-ImageNet datasets. See Table 5 for the results of these experiments. In all cases, we are able to improve the performance of existing methods, sometimes by over 5%. Even without Meta-MaxUp, we improve performance over the baseline by a large margin. The superiority of meta-learners that use these augmentation strategies suggests that data augmentation is critical for these popular algorithms and has largely been overlooked.

In addition, we compare our method to augmentation by Large Rotations at the task level – the only competing work to our knowledge – in Table 6. Note that using Large Rotations to create new classes is referred to as "Task Augmentation" in (Liu et al., 2020); we refer to it here as "Large Rotations" to avoid confusion since we study a myriad of augmentations at the task level. We observe that with the same training algorithm (MetaOptNet with SVM) and the ResNet-12 backbone, our method outperforms the Large Rotations augmentation strategy by a large margin on both the CIFAR-FS and mini-ImageNet datasets. Together with the same ensemble method as used in Large Rotations, marked by "+ens", we further boost performance consistently above the MCT baseline, the current highest performing meta-learning method on these benchmarks, despite using an older meta-learner previously thought to perform worse than MCT. Moreover, when both training and validation datasets are used for meta-training, we can achieve the state-of-art results for few-shot classification on mini-ImageNet in inductive setting.

Table 6. Few-shot classification accuracy (%) on CIFAR-FS and mini-ImageNet with ResNet-12 backbone. "M-SVM" denotes MetaOptNet with the SVM head. "+ens" denotes testing with ensemble methods as in (Liu et al., 2020). "LargeRot" denotes task-level augmentation by Large Rotations as described in (Liu et al., 2020).

	CIFAR-FS		mini-ImageNet	
Method	1-shot	5-shot	1-shot	5-shot
M-SVM + LargeRot	72.95 ± 0.24	85.91 ± 0.18	62.12 ± 0.22	78.90 ± 0.17
M-SVM + MM (ours)	75.67 ± 0.34	88.37 ± 0.23	65.02 ± 0.32	82.42 ± 0.23
M-SVM + LargeRot + ens	75.85 ± 0.24	87.73 ± 0.17	64.56 ± 0.22	81.35 ± 0.16
M-SVM + MM + ens (ours)	76.38 ± 0.33	89.16 ± 0.22	66.42 ± 0.32	83.69 ± 0.21
M-SVM + LargeRot + ens + val	**76.75 ± 0.23**	88.38 ± 0.17	65.38 ± 0.23	82.13 ± 0.16
M-SVM + MM + ens + val (ours)	76.38 ± 0.34	**89.25 ± 0.21**	**67.37 ± 0.32**	**84.57 ± 0.21**

4.7. Out-of-Distribution Testing on Meta-Dataset

In this section, we examine the effectiveness of our methods on cross-domain few-shot learning benchmarks. Few-shot learners may be successful on similar tasks to their training data but fail on tasks that deviate. Thus, testing on diverse distributions is crucial. To this end, we leverage Meta-Dataset, a collection of subdatasets used for testing meta-learners across diverse tasks (Triantafillou et al., 2019). Among the 10 subdatasets, we train the networks only on ILSVRC-2012 (Russakovsky et al., 2015), the largest dataset in the collection, and we evaluate the cross-domain few-shot classification performance on the other 9 datasets with R2-D2 and MetaOptNet learners and ResNet-12 backbones. Training and evaluation details can be found in Appendix H.

We observe that on all subdatasets except for Omniglot, our proposed methods can improve test accuracy over the baseline by as much as 7%. Additionally, we improve performance by a large margin (more than 3%) on more than half of the subdatasets. On average, Meta-MaxUp improves accuracy by around 3%. Omniglot suffers under our strategies since this dataset comprises handwritten letters which are not invariant to strong augmentations. Specially designed augmentations for handwritten letters are necessary to optimize performance on Omniglot. The success of Meta-MaxUp on cross-domain benchmarks demonstrates that the proposed strategy is effective even on diverse testing distributions which do not resemble the learner's training data.

Table 7. Few-shot classification accuracy (%) on Meta-Dataset with both MetaOptNet and R2-D2 learner. "+ DA" denotes training with CutMix (Q) + Rotation (T), and "+ MM" denotes training with Meta-MaxUp. Confidence intervals have radius equal to one standard error.

Test Source	R2-D2	+ DA	+ MM
ILSVRC	69.04 ± 0.31	70.30 ± 0.31	**71.68 ± 0.30**
Birds	75.22 ± 0.30	77.27 ± 0.28	**77.95 ± 0.30**
Omniglot	**97.46 ± 0.08**	96.10 ± 0.11	96.71 ± 0.09
Aircraft	54.28 ± 0.28	58.93 ± 0.30	**60.83 ± 0.28**
Textures	63.47 ± 0.24	65.98 ± 0.24	**67.34 ± 0.26**
Quick Draw	76.39 ± 0.27	78.44 ± 0.27	**80.83 ± 0.25**
Fungi	50.41 ± 0.22	52.29 ± 0.20	**54.12 ± 0.22**
VGG Flower	86.26 ± 0.21	87.79 ± 0.19	**90.29 ± 0.17**
Traffic Signs	83.98 ± 0.34	**84.23 ± 0.36**	83.59 ± 0.36
MSCOCO	70.29 ± 0.30	71.59 ± 0.31	**72.83 ± 0.29**

Test Source	MetaOptNet	+ DA	+ MM
ILSVRC	68.92 ± 0.30	71.17 ± 0.30	**72.19 ± 0.30**
Birds	75.58 ± 0.39	**77.49 ± 0.29**	77.47 ± 0.2
Omniglot	**97.43 ± 0.10**	95.97 ± 0.10	96.59 ± 0.09
Aircraft	53.40 ± 0.37	60.43 ± 0.29	**60.57 ± 0.29**
Textures	63.29 ± 0.33	65.70 ± 0.24	**69.42 ± 0.25**
Quick Draw	78.00 ± 0.33	79.56 ± 0.25	**80.67 ± 0.25**
Fungi	50.56 ± 0.21	53.80 ± 0.22	**53.82 ± 0.22**
VGG Flower	88.16 ± 0.25	89.92 ± 0.18	**91.13 ± 0.15**
Traffic Signs	85.12 ± 0.33	**85.25 ± 0.33**	83.38 ± 0.37
MSCOCO	69.52 ± 0.32	71.90 ± 0.31	**73.49 ± 0.30**

5. Discussion

In this work, we break down data augmentation in the context of meta-learning. In doing so, we uncover possibilities that do not exist in the classical image classification setting. We identify four modes of augmentation: query, support, task, and shot. These modes behave differently and are of varying importance. Specifically, we find that augmenting query data is particularly important. After adapting various data augmentations to meta-learning, we propose Meta-MaxUp for combining various meta-specific data augmentations. We demonstrate that Meta-MaxUp significantly improves the performance of popular meta-learning algorithms. As shown by the recent popularity of frameworks like AutoAugment (Cubuk et al., 2018) and MaxUp (Gong et al., 2020), data augmentation for standard classification is still an active area of research. We hope that this work opens up possibilities for further work on meta-specific data augmentation and that emerging methods for data augmentation will boost the performance of meta-learning on progressively larger models with more complex backbones.

Acknowledgement

This work was supported by the AFOSR MURI program, the Office of Naval Research, the DARPA YFA program, and the National Science Foundation Directorate of Mathematical Sciences. Additional support was provided by Capital One Bank and JP Morgan Chase.

References

Antoniou, A. and Storkey, A. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.

Bertinetto, L., Henriques, J. F., Torr, P. H., and Vedaldi, A. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

Chen, W.-Y., Liu, Y.-C., Kira, Z., Wang, Y.-C. F., and Huang, J.-B. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019a.

Chen, Z., Fu, Y., Chen, K., and Jiang, Y.-G. Image block augmentation for one-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3379–3386, 2019b.

Chen, Z., Fu, Y., Wang, Y.-X., Ma, L., Liu, W., and Hebert, M. Image deformation meta-networks for one-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8680–8689, 2019c.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

Gidaris, S. and Komodakis, N. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4367–4375, 2018.

Goldblum, M., Fowl, L., and Goldstein, T. Adversarially robust few-shot learning: A meta-learning approach. *arXiv*, pp. arXiv–1910, 2019.

Goldblum, M., Reich, S., Fowl, L., Ni, R., Cherepanova, V., and Goldstein, T. Unraveling meta-learning: Understanding feature representations for few-shot tasks. *arXiv preprint arXiv:2002.06753*, 2020.

Gong, C., Ren, T., Ye, M., and Liu, Q. Maxup: A simple way to improve generalization of neural network training. *arXiv preprint arXiv:2002.09024*, 2020.

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

Kumar, V., Glaude, H., de Lichy, C., and Campbell, W. A closer look at feature space data augmentation for few-shot intent classification. *arXiv preprint arXiv:1910.04176*, 2019.

Kye, S. M., Lee, H. B., Kim, H., and Hwang, S. J. Transductive few-shot learning with meta-learned confidence. *arXiv preprint arXiv:2002.12017*, 2020.

Lee, K., Maji, S., Ravichandran, A., and Soatto, S. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10657–10665, 2019.

Liu, J., Chao, F., and Lin, C.-M. Task augmentation by rotating for meta-learning. *arXiv preprint arXiv:2003.00804*, 2020.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks, 2019.

Nichol, A., Achiam, J., and Schulman, J. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Oreshkin, B., López, P. R., and Lacoste, A. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 721–731, 2018.

Qiao, S., Liu, C., Shen, W., and Yuille, A. L. Few-shot image recognition by predicting parameters from activations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7229–7238, 2018.

Rajendran, J., Irpan, A., and Jang, E. Meta-learning requires meta-augmentation. *arXiv preprint arXiv:2007.05549*, 2020.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252, 2015.

Seo, J.-W., Jung, H.-G., and Lee, S.-W. Self-augmentation: Generalizing deep networks to unseen classes for few-shot learning. *arXiv preprint arXiv:2004.00251*, 2020.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pp. 4077–4087, 2017.

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.-A., et al. Meta-dataset: A dataset of datasets for learning to learn from few examples. *arXiv preprint arXiv:1903.03096*, 2019.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Yao, H., Huang, L., Wei, Y., Tian, L., Huang, J., and Li, Z. Don't overlook the support set: Towards improving generalization in meta-learning. *arXiv preprint arXiv:2007.13040*, 2020.

Yin, M., Tucker, G., Zhou, M., Levine, S., and Finn, C. Meta-learning without memorization. *arXiv preprint arXiv:1912.03820*, 2019.

Yun, S., Han, D., Oh, S. J., Chun, S., Choe, J., and Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6023–6032, 2019.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.