DATA AUGMENTATION FOR META-LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Conventional image classifiers are trained by randomly sampling mini-batches of images. To achieve state-of-the-art performance, sophisticated data augmentation schemes are used to expand the amount of training data available for sampling. In contrast, meta-learning algorithms sample not only images, but classes as well. We investigate how data augmentation can be used not only to expand the number of images available per class, but also to generate entirely new classes. 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 related tasks, 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 the various stages of the meta-learning pipeline.

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. 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 identify places in which data augmentation can be inserted. We uncover four modes of augmentations for meta-learning: support augmentation, query augmentation, task augmentation, and shot augmentation.
- Second, we test these four modes using a pool of image augmentations, and we find that query augmentation is critical, while support augmentations often do not provide performance benefits and may even degrade accuracy in some cases.
- Third, we combine augmentations and implement a MaxUp strategy, which we call Meta-MaxUp, in order to maximize performance. We achieve significant performance boosts with popular meta-learners on both mini-ImageNet and CIFAR-FS.

2 RELATED WORK

Meta-learners are known to be particularly vulnerable to overfitting (Rajendran et al., 2020). One recent work has developed a data augmentation method to overcome this problem (Liu et al., 2020). The latter method involves simply rotating all images in a class by a large amount and considering this new rotated class to be distinct from its parent class. This effectively increases the number of possible few-shot tasks that can be sampled during training. A feature-space augmentation, MetaMix, has been proposed for averaging support features in few-shot learning (Yao et al., 2020).

A different line of work has instead applied 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).

3 THE ANATOMY OF DATA AUGMENTATION FOR META-LEARNING

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. During an episode of training, we sample a batch of tasks which may be, for example, five-way classification problems. In the inner loop, a model is fine-tuned or adapted on *support* data. Then, in the outer loop, the model is evaluated on *query* data, and we compute the gradient of the loss on the query data with respect to the model's parameters before fine-tuning. Finally, we perform a descent step, completing the episode. Intuitively, meta-learners are optimized to generalize well after fine-tuning on very little data. At test time, the model is fine-tuned on a small set of data, which is analogous to support data, and then inference is performed on other data, analogous to query data. The number of support samples per class in a few-shot classification problem is called the *shot*.

3.1 DATA AUGMENTATION MODES

We describe 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 copies of each image using data augmentation. Shot augmentation can also be used during training by adding copies of each support image via augmentation. Shot augmentation during training may better prepare meta-learners for test-time shot augmentation.

Existing meta-learning algorithms for few-shot image classification typically use horizontal flips, random crops, and color jitter on both support and query images. 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. Additionally, we find that task augmentation, when combined with query augmentation, can offer further boosts in performance when compared with existing frameworks.

3.2 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 which greatly change the image distribution are better suited for task augmentations while techniques that preserve 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 these techniques are prevalent in the literature. More details on our pool of augmentation techniques can be found in Appendix A.1.

3.3 Meta-MaxUp Augmentation for Meta-Learning

Recent work proposed 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, a set of the data augmentations, $\mathcal{S}$, collected from the four modes, as well as their combinations, is chosen. 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,

$$\min_{\theta} \mathbb{E}_{\mathcal{T}}\Big[\max_{M\in\mathcal{S}} \mathcal{L}(F_{\theta'}, M(\mathcal{T}^q))\Big], \tag{1}$$

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 $k = \arg\max_j \mathcal{L}(F_{\theta_j}, M_j(\mathcal{T}_i^q))$, 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 can further boost performance by combining augmentations with Meta-MaxUp. (Section 4.4)

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 mini-ImageNet and CIFAR-FS datasets (Vinyals et al., 2016; Bertinetto et al., 2018). 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. A description of training hyperparameters and computational complexity can be found in Appendix A.3. We report confidence intervals with a radius of one standard error.

Few-shot learning may be done 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.

4.2 An Empirical Comparison of Augmentation Modes

We empirically evaluate the performance of all four different augmentation modes identified in Section 3.1 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) and a ResNet-12 backbone. We report the results of the most effective augmentations for each modes in Table 1. The full table of results can be found in Appendix A.2.

Table 1: Few-shot classification accuracy (%) on the CIFAR-FS dataset with the most effective data augmentations for each mode shown. Confidence intervals have radius equal to one standard error. "CNN-4" denotes a 4-layer convolutional network with 96, 192, 384, and 512 filters in each layer (Bertinetto et al., 2018). Best performance in each category is bolded. Query CutMix is consistently the most effective single augmentation for meta-learning.

		CNN-4		ResNet-12	
Mode	Level	1-shot	5-shot	1-shot	5-shot
Baseline	-	67.56 ± 0.35	82.39 ± 0.26	73.01 ± 0.37	84.29 ± 0.24
Random Erase	Support	67.71 ± 0.36	82.25 ± 0.26	72.30 ± 0.37	84.50 ± 0.25
Self-Mix	Support	69.61 ± 0.35	83.43 ± 0.25	71.96 ± 0.36	84.84 ± 0.25
CutMix	Query	$\mathbf{70.54 \pm 0.33}$	$\mathbf{84.69 \pm 0.24}$	$\mathbf{75.97 \pm 0.34}$	$\mathbf{87.28 \pm 0.23}$
Random Erase	Query	69.73 ± 0.34	84.04 ± 0.25	73.05 ± 0.36	85.67 ± 0.25
Self-Mix	Query	69.54 ± 0.35	84.20 ± 0.24	73.59 ± 0.35	86.14 ± 0.24
MixUp	Task	67.21 ± 0.35	82.72 ± 0.26	72.05 ± 0.37	85.27 ± 0.25
Large Rotation	Task	68.96 ± 0.35	83.65 ± 0.25	73.79 ± 0.36	85.81 ± 0.24
Horizontal Flip	Shot	68.13 ± 0.35	82.95 ± 0.25	73.25 ± 0.36	85.06 ± 0.25
Random Crop	Shot	67.33 ± 0.36	83.04 ± 0.25	70.56 ± 0.37	83.87 ± 0.25

Table 1 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.

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 1, and we conduct experiments on the same backbones and dataset. Results are reported in Table 2. 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 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.

Table 2: Few-shot classification accuracy (%) 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.

	CNN-4		ResNet-12	
Mode	1-shot	5-shot	1-shot	5-shot
CutMix	70.54 ± 0.33	84.69 ± 0.24	75.97 ± 0.34	87.28 ± 0.23
+ CutMix (S)	69.50 ± 0.35	82.64 ± 0.26	75.00 ± 0.37	85.37 ± 0.25
+ Random Erase (S)	70.12 ± 0.35	84.48 ± 0.25	75.84 ± 0.34	87.19 ± 0.24
+ Random Erase (Q)	69.68 ± 0.34	84.36 ± 0.24	75.08 ± 0.35	87.14 ± 0.23
+ Self-Mix (S)	70.65 ± 0.34	84.68 ± 0.25	$\mathbf{76.27 \pm 0.34}$	87.52 ± 0.24
+ Self-Mix (Q)	69.94 ± 0.34	84.38 ± 0.24	76.04 ± 0.34	87.45 ± 0.24
+ MixUp (T)	70.33 ± 0.35	84.57 ± 0.25	75.97 ± 0.34	86.66 ± 0.24
+ Rotation (T)	70.35 ± 0.34	84.73 ± 0.24	75.74 ± 0.34	$\mathbf{87.68 \pm 0.24}$
+ Horizontal Flip (Shot)	$\mathbf{70.90 \pm 0.33}$	$\mathbf{84.87 \pm 0.24}$	76.23 ± 0.34	87.36 ± 0.24

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. Results are reported in Table 3, and a detailed description of the augmentation pools can be found in Appendix A.4. 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 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 used (Bertinetto et al., 2018).

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 overfit-

Table 3: Few-shot classification accuracy (%) 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.

		CNN-4		ResNet-12	
Pool	m	1-shot	5-shot	1-shot	5-shot
Baseline	-	67.56 $\pm$ 0.36	82.39 $\pm$ 0.26	73.01 $\pm$ 0.37	84.29 $\pm$ 0.24
CutMix	1	70.54 $\pm$ 0.34	84.69 $\pm$ 0.24	75.97 $\pm$ 0.34	87.28 $\pm$ 0.23
Single	1	70.76 $\pm$ 0.35	84.70 $\pm$ 0.25	75.71 $\pm$ 0.35	87.44 $\pm$ 0.43
Medium	1	70.50 $\pm$ 0.34	84.59 $\pm$ 0.24	75.60 $\pm$ 0.34	87.35 $\pm$ 0.23
Large	1	70.84 $\pm$ 0.34	85.04 $\pm$ 0.24	75.44 $\pm$ 0.34	87.47 $\pm$ 0.23
CutMix	2	70.56 $\pm$ 0.34	84.78 $\pm$ 0.24	74.93 $\pm$ 0.36	87.14 $\pm$ 0.24
Single	2	70.86 $\pm$ 0.34	85.06 $\pm$ 0.25	75.81 $\pm$ 0.34	87.33 $\pm$ 0.23
Medium	2	70.75 $\pm$ 0.34	85.02 $\pm$ 0.24	76.49 $\pm$ 0.33	88.20 $\pm$ 0.22
Large	2	70.63 $\pm$ 0.34	85.07 $\pm$ 0.24	76.59 $\pm$ 0.34	88.11 $\pm$ 0.23
CutMix	4	70.48 $\pm$ 0.34	84.76 $\pm$ 0.24	75.08 $\pm$ 0.23	87.60 $\pm$ 0.24
Single	4	**71.10** $\pm$ **0.34**	**85.50** $\pm$ **0.24**	76.82 $\pm$ 0.24	88.14 $\pm$ 0.23
Medium	4	70.58 $\pm$ 0.34	85.32 $\pm$ 0.24	76.30 $\pm$ 0.24	88.29 $\pm$ 0.22
Large	4	70.71 $\pm$ 0.34	85.04 $\pm$ 0.23	**76.99** $\pm$ **0.24**	**88.35** $\pm$ **0.22**

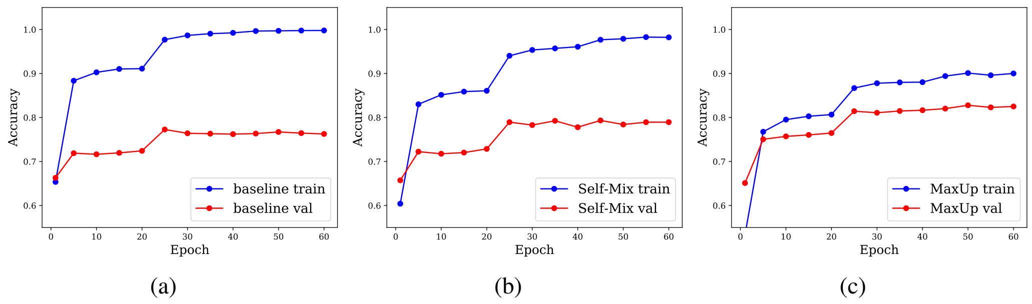


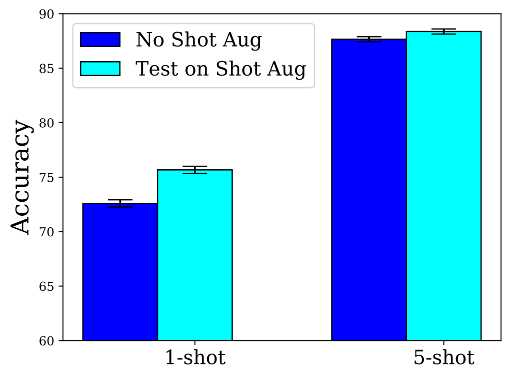
Figure 1: Training and validation accuracy for R2-D2 meta-learner with ResNet-12 backbone on the CIFAR-FS dataset. (a) Baseline model (b) query Self-Mix (c) Meta-MaxUp. Better data augmentation strategies, such as MaxUp, narrow the generalization gap and prevent overfitting.

ting 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.

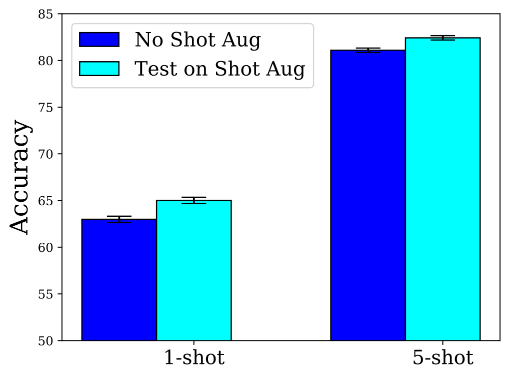
4.5 Shot Augmentation for Pre-Trained Models

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 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 on pre-trained models during testing. For additional experiments, see Appendix A.5.



(a)



(b)

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

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 4 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 results with augmentation by Large Rotations at the task level – the only competing work to our knowledge – in Table 5. Note, augmentation with 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.

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 it is particularly important to augment query data. 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 and MaxUp, 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.

Table 4: 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. "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	$\mathbf{71.10 \pm 0.34}$	$\mathbf{85.50 \pm 0.24}$	$\mathbf{58.18 \pm 0.32}$	$\mathbf{75.35 \pm 0.25}$
R2-D2	ResNet-12	73.01 ± 0.37	84.29 ± 0.24	60.46 ± 0.32	76.88 ± 0.24
+ DA	ResNet-12	76.17 ± 0.34	87.74 ± 0.24	$\mathbf{65.54 \pm 0.32}$	81.52 ± 0.23
+ MM	ResNet-12	$\mathbf{76.65 \pm 0.33}$	$\mathbf{88.57 \pm 0.24}$	65.15 ± 0.32	$\mathbf{81.76 \pm 0.24}$
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	$\mathbf{50.38 \pm 0.32}$	$\mathbf{71.44 \pm 0.26}$
+ MM	64-64-64-64	$\mathbf{63.01 \pm 0.36}$	$\mathbf{80.85 \pm 0.25}$	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	$\mathbf{76.05 \pm 0.34}$	$\mathbf{87.84 \pm 0.23}$	$\mathbf{62.81 \pm 0.34}$	$\mathbf{79.38 \pm 0.24}$
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	$\mathbf{75.67 \pm 0.34}$	$\mathbf{88.37 \pm 0.23}$	$\mathbf{65.02 \pm 0.32}$	$\mathbf{82.42 \pm 0.23}$
MCT	ResNet-12	75.80 ± 0.33	89.10 ± 0.42	64.84 ± 0.33	81.45 ± 0.23
+ MM	ResNet-12	$\mathbf{76.00 \pm 0.33}$	$\mathbf{89.54 \pm 0.33}$	$\mathbf{66.37 \pm 0.32}$	$\mathbf{83.11 \pm 0.22}$

Table 5: 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 + LargeRot + ens	75.85 ± 0.24	87.73 ± 0.17	64.56 ± 0.22	81.35 ± 0.16
M-SVM + MM (ours)	75.67 ± 0.34	88.37 ± 0.23	65.02 ± 0.32	82.42 ± 0.23
M-SVM + MM + ens (ours)	$\mathbf{76.38 \pm 0.33}$	$\mathbf{89.16 \pm 0.22}$	$\mathbf{66.42 \pm 0.32}$	$\mathbf{83.69 \pm 0.21}$

REFERENCES

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

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019a.

Zitian Chen, Yanwei Fu, Kaiyu Chen, and Yu-Gang Jiang. Image block augmentation for one-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3379–3386, 2019b.

Zitian Chen, Yanwei Fu, Yu-Xiong Wang, Lin Ma, Wei Liu, and Martial Hebert. Image deformation meta-networks for one-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8680–8689, 2019c.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Micah Goldblum, Liam Fowl, and Tom Goldstein. Adversarially robust few-shot learning: A meta-learning approach. *arXiv*, pp. arXiv–1910, 2019.

Micah Goldblum, Steven Reich, Liam Fowl, Renkun Ni, Valeriia Cherepanova, and Tom Goldstein. Unraveling meta-learning: Understanding feature representations for few-shot tasks. *arXiv preprint arXiv:2002.06753*, 2020.

Chengyue Gong, Tongzheng Ren, Mao Ye, and Qiang Liu. Maxup: A simple way to improve generalization of neural network training. *arXiv preprint arXiv:2002.09024*, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Varun Kumar, Hadrien Glaude, Cyprien de Lichy, and William Campbell. A closer look at feature space data augmentation for few-shot intent classification. *arXiv preprint arXiv:1910.04176*, 2019.

Seong Min Kye, Hae Beom Lee, Hoirin Kim, and Sung Ju Hwang. Transductive few-shot learning with meta-learned confidence. *arXiv preprint arXiv:2002.12017*, 2020.

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

Jialin Liu, Fei Chao, and Chih-Min Lin. Task augmentation by rotating for meta-learning. *arXiv preprint arXiv:2003.00804*, 2020.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 721–731, 2018.

Janarthanan Rajendran, Alex Irpan, and Eric Jang. Meta-learning requires meta-augmentation. *arXiv preprint arXiv:2007.05549*, 2020.

Jin-Woo Seo, Hong-Gyu Jung, and Seong-Whan Lee. Self-augmentation: Generalizing deep networks to unseen classes for few-shot learning. *arXiv preprint arXiv:2004.00251*, 2020.

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

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Huaxiu Yao, Longkai Huang, Ying Wei, Li Tian, Junzhou Huang, and Zhenhui Li. Don't overlook the support set: Towards improving generalization in meta-learning. *arXiv preprint arXiv:2007.13040*, 2020.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. *arXiv preprint arXiv:1912.03820*, 2019.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

A Appendix

A.1 Details About Data Augmentation Techniques

In this section, we provide more details about the different data augmentation techniques we use in this work. We employ the following pool of data augmentation techniques:

CutMix: Yun et al. (2019) introduced the CutMix augmentation strategy where patches are cut and pasted among training images, and the ground truth labels are also mixed proportionally to the area of the patches.

MixUp: Zhang et al. (2017) proposed mixup, a simple learning principle to alleviate memorization and sensitivity to adversarial examples. Mixup trains a neural network on convex combinations of pairs of examples and their labels. By doing so, mixup regularizes the neural network to favor simple linear behavior in between training examples.

Self-Mix: Seo et al. (2020) introduced the self-mix augmentation strategy in which a patch of an image is substituted into other values in the same image to improve the generalization ability of few-shot image classification models.

In addition, we use some standard and simple data augmentation techniques:

Rotation: augments the data by rotating the images.

Horizontal Flip: augments the data by horizontally flipping images.

Random Erase: augments the data by randomly erasing patches from the image.

Finally, we also experimented with the following data augmentation techniques:

Combining Labels: augments the data by combining two different labels into a single class. For instance, we combine may combine the "dog" and "cat" labels to create a new "dog or cat" class.

Feature Mixup: similar to the "Mixup" augmentation technique we describe above, however we perform the mixup strategy on the feature representation for the image.

Drop Channel: augments the data by dropping color channels in the image.

Solarize: inverts all pixels above a threshold value of magnitude.

A.2 Results for All Data Augmentation Techniques

Table 6: Few-shot classification accuracy (%) on the CIFAR-FS dataset for all data augmentations. Confidence intervals have radius equal to one standard error. "CNN-4" denotes a 4-layer convolutional network with 96, 192, 384, and 512 filters in each layer (Bertinetto et al., 2018). Best performance in each category is bolded.

		CNN-4		ResNet-12	
Mode	Level	1-shot	5-shot	1-shot	5-shot
Baseline	-	67.56 ± 0.35	82.39 ± 0.26	73.01 ± 0.37	84.29 ± 0.24
Random Erase	Support	67.71 ± 0.36	82.25 ± 0.26	72.30 ± 0.37	84.50 ± 0.25
Self-Mix	Support	**69.61 ± 0.35**	**83.43 ± 0.25**	**71.96 ± 0.36**	**84.84 ± 0.25**
CutMix	Support	66.09 ± 0.36	80.34 ± 0.27	70.50 ± 0.39	81.30 ± 0.28
MixUp	Support	66.13 ± 0.37	79.49 ± 0.27	69.65 ± 0.38	82.02 ± 0.26
Feature Mixup	Support	67.88 ± 0.35	82.40 ± 0.25	71.21 ± 0.37	83.38 ± 0.25
Rotation	Support	68.65 ± 0.35	82.86 ± 0.25	71.13 ± 0.37	83.84 ± 0.25
Combining labels	Support	68.27 ± 0.36	82.53 ± 0.26	71.00 ± 0.38	83.12 ± 0.25
Drop Channel	Support	68.21 ± 0.35	82.76 ± 0.25	69.65 ± 0.73	83.15 ± 0.25
Solarize	Support	68.65 ± 0.35	82.68 ± 0.26	70.88 ± 0.37	83.45 ± 0.25
Random Erase	Query	69.73 ± 0.34	84.04 ± 0.25	73.05 ± 0.36	85.67 ± 0.25
Self-Mix	Query	69.61 ± 0.35	83.43 ± 0.25	71.96 ± 0.36	84.84 ± 0.25
CutMix	Query	**70.54 ± 0.33**	**84.69 ± 0.24**	**75.97 ± 0.34**	**87.28 ± 0.23**
MixUp	Query	67.70 ± 0.34	83.13 ± 0.25	72.93 ± 0.35	86.13 ± 0.24
Feature Mixup	Query	70.16 ± 0.35	83.80 ± 0.28	73.38 ± 0.35	85.87 ± 0.23
Rotation	Query	68.17 ± 0.35	83.01 ± 0.25	72.02 ± 0.36	84.42 ± 0.25
Combining labels	Query	66.01 ± 0.34	81.99 ± 0.26	69.77 ± 0.37	82.99 ± 0.26
Drop Channel	Query	68.34 ± 0.35	83.25 ± 0.25	69.60 ± 0.37	83.01 ± 0.26
Solarize	Query	67.51 ± 0.35	82.65 ± 0.25	72.45 ± 0.36	84.97 ± 0.24
MixUp	Task	67.21 ± 0.35	82.72 ± 0.26	72.05 ± 0.37	85.27 ± 0.25
Large Rotation	Task	**68.96 ± 0.35**	**83.65 ± 0.25**	**73.79 ± 0.36**	**85.81 ± 0.24**
CutMix	Task	68.78 ± 0.36	82.99 ± 0.50	72.72 ± 0.37	84.62 ± 0.25
Combining labels	Task	68.08 ± 0.35	82.33 ± 0.26	69.64 ± 0.37	83.79 ± 0.26
Random Erase	Task	68.39 ± 0.36	83.26 ± 0.25	71.09 ± 0.37	84.49 ± 0.25
Drop Channel	Task	67.54 ± 0.36	81.97 ± 0.25	70.24 ± 0.37	83.52 ± 0.26
Horizontal Flip	Shot	**68.13 ± 0.35**	82.95 ± 0.25	**73.25 ± 0.36**	**85.06 ± 0.25**
Random Crop	Shot	67.33 ± 0.36	**83.04 ± 0.25**	70.56 ± 0.37	83.87 ± 0.25
Random Rotation	Shot	67.57 ± 0.35	83.00 ± 0.25	70.32 ± 0.37	83.75 ± 0.25

A.3 Training Details

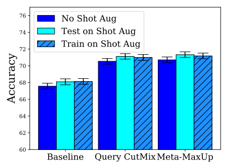
For MetOptNet, we use the same training procedure as (Lee et al., 2019) including SGD with Nesterov momentum of 0.9 and weight decay coefficient 0.0005. The model was meta-trained for 60 epochs, with an initial learning rate 0.1, then changed to 0.006, 0.0012, and 0.00024 at epochs 20, 40 and 50, respectively. In each epoch, we train on 8000 episodes and use mini-batches of size 8. Following (Lee et al., 2019), we use a larger shot number (15) to train mini-ImageNet for both 1-shot and 5-shot classification. For MCT, we use the same optimizer but with batch size 1 and maximum iterations 50000. Following (Kye et al., 2020), we enlarge the training classification ways to 15 for a 5-way testing. We use instance-wise metric for all inductive learning.

A.4 Augmentation Pool for Meta-MaxUp

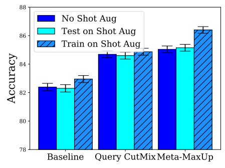
For all the benchmark results of Meta-MaxUp training, we use a medium-size data augmentation pool with $m = 4$, including CutMix (Q), Random Erase (Q), Self-Mix (S), Rotation (T), CutMix (Q) + Rotation (T), and Random Erase (Q) + Rotation (T). For the large-size pool, we add more techniques and combinations of the mentioned techniques into the pool, including Random Erase

(Q) + Random Erase (S), CutMix (Q) + Random Erase (S), CutMix (Q) + Random Erase (Q), and CutMix (Q) + Self-Mix (S).

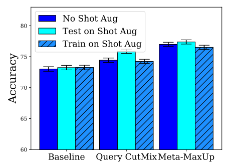
A.5 Bar Plots for Shot Augmentation



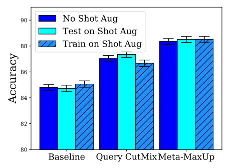
(a)



(b)



(c)



(d)

Figure 3: Performance for shot augmentation for different backbone works and training strategies on CIFAR-FS. (a) 1-shot classification for CNN-4 (b) 5-shot classification for CNN-4 (c) 1-shot classification for ResNet-12 (d) 5-shot classification for ResNet-12