Improving Generalization in Meta-learning via Task Augmentation

Huaxiu Yao[† 1] **Long-Kai Huang**[2] **Linjun Zhang**[3] **Ying Wei**[4] **Li Tian**[2]
James Zou[1] **Junzhou Huang**[2] **Zhenhui Li**[5]

Abstract

Meta-learning has proven to be a powerful paradigm for transferring the knowledge from previous tasks to facilitate the learning of a novel task. Current dominant algorithms train a well-generalized model initialization which is adapted to each task via the support set. The crux lies in optimizing the generalization capability of the initialization, which is measured by the performance of the adapted model on the query set of each task. Unfortunately, this generalization measure, evidenced by empirical results, pushes the initialization to overfit the meta-training tasks, which significantly impairs the generalization and adaptation to novel tasks. To address this issue, we actively augment a meta-training task with "more data" when evaluating the generalization. Concretely, we propose two task augmentation methods, including MetaMix and Channel Shuffle. MetaMix linearly combines features and labels of samples from both the support and query sets. For each class of samples, Channel Shuffle randomly replaces a subset of their channels with the corresponding ones from a different class. Theoretical studies show how task augmentation improves the generalization of meta-learning. Moreover, both MetaMix and Channel Shuffle outperform state-of-the-art results by a large margin across many datasets and are compatible with existing meta-learning algorithms.

1. Introduction

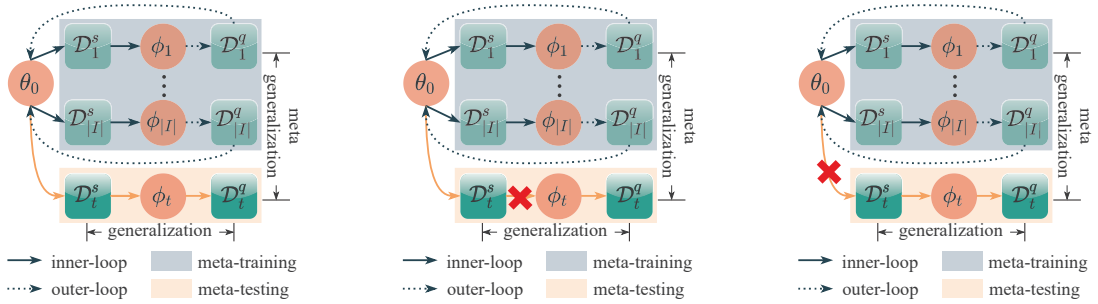
Meta-learning, or learning to learn (Thrun & Pratt, 1998), empowers agents with the core aspect of intelligence– quickly learning a new task with as little as a few examples by drawing upon the knowledge learned from prior tasks. The resurgence of meta-learning recently pushes ahead with more effective algorithms that have been deployed in areas such as computer vision (Kang et al., 2019; Liu et al., 2019; Sung et al., 2018), natural language processing (Dou et al., 2019; Gu et al., 2018; Madotto et al., 2019), and robotics (Xie et al., 2018; Yu et al., 2018). Some of the dominant algorithms learn a transferable metric space from previous tasks (Snell et al., 2017; Vinyals et al., 2016), unfortunately being only applicable to classification problems. Instead, gradient-based algorithms (Finn et al., 2017; 2018) framing meta-learning as a bi-level optimization problem are flexible and general enough to be independent of problem types, which we focus on in this work.

The bi-level optimization procedure of gradient-based algorithms is illustrated in Figure 1a. In the inner-loop, the initialization of a base model (a.k.a., base learner) globally shared across tasks (i.e., θ_0) is adapted to each task (e.g., ϕ_1 for the first task) via gradient descent over the support set of the task. To reach the desired goal that optimizing from this initialization leads to fast adaptation and generalization, a meta-training objective evaluating the generalization capability of the initialization on all meta-training tasks is optimized in the outer-loop. Specifically, the generalization capability on each task is measured by the performance of the adapted model on a set distinguished from the support, namely the query set.

The learned initialization, however, is at high risk of two forms of overfitting: (1) *memorization overfitting* (Yin et al., 2020) (Figure 1b) where it solves meta-training tasks via rote memorization and does not rely on support sets for inner-loop adaptation and (2) *learner overfitting* (Rajendran et al., 2020) (Figure 1c) where it overfits to the meta-training tasks and fails to generalize to the meta-testing tasks though support sets come into play during inner-loop adaptation. Both types of overfitting hurt the generalization from meta-training to meta-testing tasks, which we call meta-generalization in Figure 1a. Improving the meta-generalization is especially challenging – standard regularizers like weight decay lose their power as they limit the flexibility of fast adaptation in the inner-loop.

[†]Part of the work was done when H.Y. was a student at Penn State University. H.Y. and LK.H. contribute equally. [1]Stanford University, CA, USA [2]Tencent AI Lab, Shenzhen, China [3]Rutgers University, NJ, USA [4]City University of Hong Kong, Hong Kong [5]Pennsylvania State University, PA, USA. Correspondence to: Ying Wei <yingwei@cityu.edu.hk>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.



(a) : Gradient-based Meta-learning (b) : Memorization Overfitting (c) : Learner Overfitting

Figure 1. (a) Illustration of the gradient-based meta-learning process and two types of generalization; (b)&(c) Two forms of overfitting in gradient-based meta-learning. The red cross represents where the learned knowledge can not be well-generalized.

To this end, the few existing solutions attempt to regularize the search space of the initialization (Yin et al., 2020) or enforce a fair performance of the initialization across all meta-training tasks (Jamal & Qi, 2019) while preserving the expressive power for adaptation. Rather than passively imposing regularization on the initialization, recently, Rajendran et al. (2020) turned towards an active data augmentation way, aiming to anticipate more data to meta-train the initialization by injecting the same noise to the labels of both support and query sets (i.e., label shift). Though the label shift with a random constant increases the dependence of the base learner on the support set, learning the constant is as easy as modifying a bias. Therefore, little extra knowledge is introduced to meta-train the initialization.

This paper sets out to investigate more flexible and powerful ways to produce "more" data via task augmentation. The goal for task augmentation is to increase the dependence of target predictions on the support set and provide additional knowledge to optimize the model initialization. To meet the goal, we propose two task augmentation strategies – **MetaMix** and **Channel Shuffle**. MetaMix linearly combines either original features or hidden representations of the support and query sets, and performs the same linear interpolation between their corresponding labels. For classification problems, MetaMix is further enhanced by the strategy of Channel Shuffle, which is named as MMCF. For samples of each class, Channel Shuffle randomly selects a subset of channels to replace with corresponding ones of samples from a different class. These additional signals for the meta-training objective improve the meta-generalization of the learned initialization as expected.

We would highlight the primary contributions of this work. (1) We identify and formalize effective task augmentation that is sufficient for alleviating both memorization overfitting and learner overfitting and thereby improving meta-generalization, resulting in two task augmentation methods. (2) Both task augmentation strategies have been theoretically proved to indeed improve the meta-generalization. (3) Throughout comprehensive experiments, we demonstrate two significant benefits of the two augmentation strategies. First, in various real-world datasets, the performances are substantially improved over state-of-the-art meta-learning algorithms and other strategies for overcoming overfitting (Jamal & Qi, 2019; Yin et al., 2020). Second, both MetaMix and MMCF are compatible with existing and advanced meta-learning algorithms and ready to boost their performances.

2. Preliminaries

Gradient-based meta-learning algorithms assume a set of tasks to be sampled from a distribution $p(\mathcal{T})$. Each task $\mathcal{T}_i$ consists of a support sample set $\mathcal{D}_i^s = \{(\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s)\}_{j=1}^{K^s}$ and a query sample set $\mathcal{D}_i^q = \{(\mathbf{x}_{i,j}^q, \mathbf{y}_{i,j}^q)\}_{j=1}^{K^q}$, where K^s and K^q denote the number of source and query samples, respectively. The objective of meta-learning is to master new tasks quickly by adapting a well-generalized model learned over the task distribution $p(\mathcal{T})$. Specifically, the model f parameterized by θ is trained on massive tasks sampled from $p(\mathcal{T})$ during meta-training. When it comes to meta-testing, f is adapted to a new task $\mathcal{T}_t$ with the help of the support set $\mathcal{D}_t^s$ and evaluated on the query set $\mathcal{D}_t^q$.

Take model-agnostic meta-learning (MAML) (Finn et al., 2017) as an example. The well-generalized model is grounded to an initialization for f, i.e., θ_0, which is adapted to each i-th task in a few gradient steps by its support set $\mathcal{D}_i^s$. The generalization performance of the adapted model, i.e., ϕ_i, is measured on the query set $\mathcal{D}_i^q$, and in turn used to optimize the initialization θ_0 during meta-training. Let $\mathcal{L}$ and μ denote the loss function and the inner-loop learning rate, respectively. The above interleaved process is formulated as a bi-level optimization problem,

$$\begin{aligned} &\theta_0^* := \min_{\theta_0} \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \left[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i^q), \mathbf{Y}_i^q) \right], \\ &\text{s.t. } \phi_i = \theta_0 - \mu \nabla_{\theta_0} \mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s), \end{aligned} \tag{1}$$

where $\mathbf{X}_i^{s(q)}$ and $\mathbf{Y}_i^{s(q)}$ represent the collection of samples and their corresponding labels for the support (query) set, respectively. The predicted value $f_{\phi_i}(\mathbf{X}_i^{s(q)})$ is denoted as $\hat{\mathbf{Y}}_i^{s(q)}$. In the meta-testing phase, to solve the new task $\mathcal{T}_t$,

the optimal initialization θ_0^* is fine-tuned on its support set $\mathcal{D}_t^s$ to the resulting task-specific parameters ϕ_t.

3. Task Augmentation

In practical situations, the distribution $p(\mathcal{T})$ is unknown for estimation of the expected performance in Eqn. (1). Instead, the common practice is to approximate it with the empirical performance, i.e.,

$$\begin{aligned}\theta_0^* := \min_{\theta_0} \frac{1}{n_T}\sum_{i=1}^{n_T}\left[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i^q), \mathbf{Y}_i^q)\right], \\ \text{s.t. } \phi_i = \theta_0 - \mu\nabla_{\theta_0}\mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s).\end{aligned} \tag{2}$$

Unfortunately, this empirical risk observes the generalization ability of the initialization θ_0 only at a finite set of n_T tasks. When the function f is sufficiently powerful, a trivial solution of θ_0 is to overfit all tasks. Compared to standard supervised learning, the overfitting is more complicated with two cases: *memorization overfitting* and *learner overfitting* which differ primarily in whether the support set contributes to inner-loop adaptation. In memorization overfitting, θ_0^* memorizes all tasks, so that the adaptation to each task via its support set is even futile (Yin et al., 2020). In learner overfitting, θ_0^* fails to generalize to new tasks, though it adapts to solve each meta-training task sufficiently with the corresponding support set (Rajendran et al., 2020). Both overfitting lead to poor meta-generalization (see Figure 1a).

Inspired by data augmentation (Cubuk et al., 2019; Zhang et al., 2018; Zhong et al., 2020; Zhang et al., 2021) which is used to mitigate the overfitting of training samples in conventional supervised learning, we propose to alleviate the problem of task overfitting via task augmentation. Before proceeding to our solutions, we first formally define two criteria for an effective task augmentation as:

Definition 1 (Criteria of Effective Task Augmentation) *An effective task augmentation for meta-learning is an augmentation function $g(\cdot)$ that transforms a task $\mathcal{T}_i = \{\mathcal{D}_i^s, \mathcal{D}_i^q\}$ to an augmentated task $\mathcal{T}_i^{'} = \{g(\mathcal{D}_i^s), g(\mathcal{D}_i^q)\}$, so that the following two criteria are met:*

(1) $I(g(\hat{\mathbf{Y}}_i^q); g(\mathcal{D}_i^s)|\theta_0, g(\mathbf{X}_i^q)) - I(\hat{\mathbf{Y}}_i^q; \mathcal{D}_i^s|\theta_0, \mathbf{X}_i^q) > 0$,

(2) $I(\theta_0; g(\mathcal{D}_i^q)|\mathcal{D}_i^q) > 0$.

The augmented task satisfying the first criterion is expected to alleviate the memorization overfitting, as the model more heavily relies on the support set $\mathcal{D}_i^s$ to make predictions, i.e., increasing the mutual information between $g(\hat{\mathbf{Y}}_i^q)$ and $g(\mathcal{D}_i^s)$. The second criterion guarantees that the augmented task contributes additional knowledge to update the initialization in the outer-loop. With more augmented meta-training tasks satisfying this criterion, the meta-generalization ability of the initialization to meta-testing tasks improves. Building on this, we will introduce the proposed task augmentation strategies.

MetaMix. One of the most immediate choices for task augmentation is directly incorporating support sets in the outer-loop, while it is far from enough. The support sets contribute little to the value and gradients of the meta-training objective, as the meta-training objective is formulated as the performance of the adapted model which is exactly optimized via support sets. Thus, we are motivated to produce "more" data out of the accessible support and query sets, resulting in MetaMix, which meta-trains θ_0 by mixing samples from both the query set and the support set.

In detail, the strategy of mixing follows Manifold Mixup (Verma et al., 2019) where not only inputs but also hidden representations are mixed up. Assume that the model f consists of L layers. The hidden representation of a sample set $\mathbf{X}$ at the l-th layer is denoted as $f_{\theta^l}(\mathbf{X})$ $(0 \leq l \leq L-1)$, where $f_{\theta^0}(\mathbf{X}) = \mathbf{X}$. For a pair of support and query sets with their corresponding labels in the i-th task $\mathcal{T}_i$, i.e., $(\mathbf{X}_i^s, \mathbf{Y}_i^s)$ and $(\mathbf{X}_i^q, \mathbf{Y}_i^q)$, we randomly sample a value of $l \in \mathcal{C} = \{0, 1, \cdots, L-1\}$ and compute the mixed batch of data for meta-training as,

$$\begin{aligned}\mathbf{X}_{i,l}^{mix} &= \boldsymbol{\lambda} f_{\phi_i^l}(\mathbf{X}_i^s) + (\mathbf{I} - \boldsymbol{\lambda}) f_{\phi_i^l}(\mathbf{X}_i^q), \\ \mathbf{Y}_i^{mix} &= \boldsymbol{\lambda}\mathbf{Y}_i^s + (\mathbf{I} - \boldsymbol{\lambda})\mathbf{Y}_i^q,\end{aligned} \tag{3}$$

where $\boldsymbol{\lambda} = \text{diag}(\{\lambda_j\}_{j=1}^{K^q})$ and each coefficient $\lambda_j \sim \text{Beta}(\alpha, \beta)$. Here, we assume that the size of the support set and that of the query are equal, i.e., $K^s = K^q$. If $K^s < K^q$, for each sample in the query set, we randomly select one sample from the support set for mixup. The similar sampling applies to $K^s > K^q$. In Appendix B.1, we illustrate the Beta distribution in both symmetric (i.e., $\alpha = \beta$) and skewed shapes (i.e., $\alpha \neq \beta$). Using the mixed batch by MetaMix, we reformulate the outer-loop optimization problem as,

$$\theta_0^* := \min_{\theta_0} \frac{1}{n_T}\sum_{i=1}^{n_T} \mathbb{E}_{\boldsymbol{\lambda}\sim\text{Beta}(\alpha,\beta)}\mathbb{E}_{l\sim\mathcal{C}}[\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})], \tag{4}$$

where $f_{\phi_i^{L-l}}$ represents the rest of layers after the mixed layer l. MetaMix is flexible enough to be compatible with off-the-shelf gradient-based meta-learning algorithms, by replacing the query set with the mixed batch for meta-training. Further, to verify the effectiveness of MetaMix, we examine whether the criteria in Definition 1 are met in the follows.

Corollary 1 *Assume that the support set is sampled independently from the query set. Then the following two equations hold:*

$$\begin{aligned}&I(\hat{\mathbf{Y}}_i^{mix}; (\mathbf{X}_i^s, \mathbf{Y}_i^s)|\theta_0, \mathbf{X}_i^{mix}) - I(\hat{\mathbf{Y}}_i^q; (\mathbf{X}_i^s, \mathbf{Y}_i^s)|\theta_0, \mathbf{X}_i^q) \\ =&H(\hat{\mathbf{Y}}_i^s|\theta_0, \mathbf{X}_i^s) \geq 0; \\ &I(\theta_0; \mathbf{X}_i^{mix}, \mathbf{Y}_i^{mix}|\mathbf{X}_i^q, \mathbf{Y}_i^q) = H(\theta_0) - H(\theta_0|\mathbf{X}_i^s, \mathbf{Y}_i^s).\end{aligned} \tag{5}$$

The first criterion is easily satisfied – $H(\hat{\mathbf{Y}}_i^s|\theta_0, \mathbf{X}_i^s)$ hardly equals zero as θ_0 unlikely fits the support set in meta-learning. The second criterion indicates that MetaMix contributes a novel task as long as the support set of the task being augmented is capable of reducing the uncertainty of the initialization θ_0, which is often the case. We provide the detailed proof of Corollary 1 in Appendix A.1.

MetaMix enhanced with Channel Shuffle. In classification, the proposed MetaMix can be further enhanced by another task augmentation strategy named Channel Shuffle (CF). Channel Shuffle aims to randomly replace a subset of channels through samples of each class by the corresponding ones in a different class. Assume that the hidden representation $f_{\phi_i^l}(\mathbf{x}_{i,j}^{s(q)})$ of each sample consists of p channels, i.e., $f_{\phi_i^l}(\mathbf{x}_{i,j}^{s(q)}) = [f_{\phi_i^l}^{(1)}(\mathbf{x}_{i,j}^{s(q)}); \ldots; f_{\phi_i^l}^{(p)}(\mathbf{x}_{i,j}^{s(q)})]$. Provided with 1) a pair of classes c and c' with corresponding sample sets $(\mathbf{X}_{i;c}^{s(q)}, \mathbf{Y}_{i;c}^{s(q)})$, $(\mathbf{X}_{i;c'}^{s(q)}, \mathbf{Y}_{i;c'}^{s(q)})$, and 2) a random variable $\mathbf{R}_{c,c'} = \text{diag}(r_1, ..., r_p)$ with $r_t \sim \text{Bernoulli}(\delta)$ and $\delta > 0.5$ for $t \in [p]$, the channel shuffle process is formulated as:

$$\begin{aligned} \mathbf{X}_{i;c}^{s(q),cf} &= \mathbf{R}_{c,c'} f_{\phi_i^l}(\mathbf{X}_{i;c}^{s(q)}) + (\mathbf{I} - \mathbf{R}_{c,c'}) f_{\phi_i^l}(\mathbf{X}_{i;c'}^{s(q)}), \\ \mathbf{Y}_{i;c}^{s(q),cf} &= \mathbf{Y}_{i;c}^{s(q)}. \end{aligned} \tag{6}$$

The channel shuffle strategy is then applied in both support and query sets, with $\mathbf{R}_{c,c'}$ shared between the two sets. We denote the shuffled support set and query set as $(\mathbf{X}_i^{s,cf}, \mathbf{Y}_i^{s,cf})$ and $(\mathbf{X}_i^{q,cf}, \mathbf{Y}_i^{q,cf})$, respectively. Then, in the outer-loop, the channel shuffled samples will be integrated into the MetaMix and the Eqn. (4) is reformulated as:

$$\begin{aligned} \mathbf{X}_{i,l}^{mmcf} &= \boldsymbol{\lambda} \mathbf{X}_i^{s,cf} + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{X}_i^{q,cf}, \\ \mathbf{Y}_i^{mmcf} &= \boldsymbol{\lambda} \mathbf{Y}_i^{s,cf} + (\mathbf{I} - \boldsymbol{\lambda}) \mathbf{Y}_i^{q,cf}, \end{aligned} \tag{7}$$

We name the MetaMix enhanced with channel shuffle as MMCF. In Appendix A.2, we prove that MMCF not only meets the first criterion in Definition 1, but also outperforms MetaMix regarding the second criterion. Taking MAML as an example, we show MetaMix and MMCF in Alg. 1 and Appendix B.2, respectively.

4. Theoretic Analysis

In this section, we theoretically investigate how the proposed task augmentation methods improve generalization, by analyzing the following two-layer neural network model. For each task $\mathcal{T}_i$, we consider minimizing the squared loss $\mathcal{L}(f_{\phi_i}(\mathbf{X}_i), \mathbf{Y}_i) = (f_{\phi_i}(\mathbf{X}_i) - \mathbf{Y}_i)^2$ with f_{ϕ_i} modeled by

$$f_{\phi_i}(\mathbf{X}_i) = \phi_i^\top \sigma(\mathbf{W}\mathbf{X}_i), \tag{8}$$

where ϕ_i is the task adapted parameters and $\mathbf{W}$ is the global shared parameter. Note that, the formulation of function f is the equivalent to ANIL (Raghu et al., 2020) under the two-layer neural network scenario, where only the head layer is adapted during the inner-loop. In the following, we will detail the analysis of MetaMix and Channel Shuffle.

Algorithm 1 Meta-training Process of MAML-MetaMix

Require: Task distribution $p(\mathcal{T})$; Learning rate μ, η; Beta distribution parameters α, β; MetaMix candidate layer set $\mathcal{C}$

Randomly initialize parameter θ_0
while not converge **do**
  Sample a batch of tasks $\{\mathcal{T}_i\}_{i=1}^n$
  for all $\mathcal{T}_i$ **do**
    Sample support set $\mathcal{D}_i^s = \{(\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s)\}_{j=1}^{K^s}$ and query set $\mathcal{D}_i^q = \{(\mathbf{x}_{i,j}^q, \mathbf{y}_{i,j}^q)\}_{j=1}^{K^q}$ from $\mathcal{T}_i$
    Compute the task-specific parameter ϕ_i via the inner-loop gradient descent, i.e., $\phi_i = \theta_0 - \mu \nabla_{\theta_0} \mathcal{L}(f_{\theta_0}(\mathbf{X}_i^s), \mathbf{Y}_i^s)$
    Sample MetaMix parameter $\boldsymbol{\lambda} \sim \text{Beta}(\alpha, \beta)$ and mixed layer l from $\mathcal{C}$
    Forward both support and query sets and mixed them at layer l as: $\mathbf{X}_{i,l}^{mix} = \boldsymbol{\lambda} f_{\phi_i^l}(\mathbf{X}_i^s) + (\mathbf{I} - \boldsymbol{\lambda}) f_{\phi_i^l}(\mathbf{X}_i^q)$, $\mathbf{Y}_i^{mix} = \boldsymbol{\lambda}\mathbf{Y}_i^s + (\mathbf{I} - \boldsymbol{\lambda})\mathbf{Y}_i^q$
    Continual forward $\mathbf{X}_{i,l}^{mix}$ to the rest of layers and compute the loss as $\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})$
  end for
  Update $\theta_0 \leftarrow \theta_0 - \eta \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\boldsymbol{\lambda} \sim \text{Beta}(\alpha,\beta)} \mathbb{E}_{l \sim \mathcal{C}} [\mathcal{L}(f_{\phi_i^{L-l}}(\mathbf{X}_{i,l}^{mix}), \mathbf{Y}_i^{mix})]$
end while

Analysis of MetaMix. In the analysis of MetaMix, we consider a symmetric version of MetaMix algorithm for technical reasons. Empirically we find that this symmetric version and the proposed MetaMix algorithm generate mostly identical results (see Appendix C for details). Specifically, for each task $\mathcal{T}_i$, we denote $\mathbf{Z}_i = \{\mathbf{x}_{i,j}, \mathbf{y}_{i,j}\}_{j=1}^{K^m} = \{\mathbf{x}_{i,j}^s, \mathbf{y}_{i,j}^s\}_{j=1}^{K^s} \cup \{\mathbf{x}_{i,j}^q, \mathbf{y}_{i,j}^q\}_{j=1}^{K^q}$, and $K^m = K^s + K^q$. Further, we consider the following MetaMix algorithm trains the second layer parameter ϕ_i by minimizing the squared loss on the mixed batch of data $\mathbf{Z}_i^{mix} = \{\mathbf{x}_{i,j}^{mix}, \mathbf{y}_{i,j}^{mix}\}_{j=1}$, where $\mathbf{Z}_i^{mix}$ is constructed as

$$\begin{aligned} \mathbf{x}_{i,j}^{mix} &= \lambda \sigma(\mathbf{W}\mathbf{x}_{i,j}) + (1-\lambda)\sigma(\mathbf{W}\mathbf{x}_{i,j'}), \\ \mathbf{y}_{i,j}^{mix} &= \lambda \mathbf{y}_{i,j} + (1-\lambda)\mathbf{y}_{i,j'}, \end{aligned} \tag{9}$$

where j' is a uniform sample from $[K^m]$ and $\lambda \sim \text{Beta}(\alpha, \beta)$.

Extending the analysis in (Zhang et al., 2021), we have the following theorem on the second-order approximation of $\mathcal{L}(\mathbf{Z}_i^{mix})$.

Lemma 1 *Consider the model set-up described above with mixup distribution $\lambda \sim \text{Beta}(\alpha, \beta)$. Then the second-order approximation of $\mathcal{L}(\mathbf{Z}_i^{mix})$ is given by*

$$\mathcal{L}(\mathbf{Z}_i) + c \cdot \phi_i^\top \Big(\frac{1}{K^m} \sum_{j=1}^{K^m} \sigma(\mathbf{W}\mathbf{x}_{i,j}) \sigma(\mathbf{W}\mathbf{x}_{i,j})^\top\Big) \phi_i, \tag{10}$$

where $c = \mathbb{E}_{\mathcal{D}_\lambda}[\frac{(1-\lambda)^2}{2\lambda^2}]$ *with* $\mathcal{D}_\lambda \sim \frac{\alpha}{\alpha+\beta}\text{Beta}(\alpha+1,\beta) + \frac{\beta}{\alpha+\beta}\text{Beta}(\beta+1,\alpha)$.

This result suggests that the MetaMix algorithm is imposing a quadratic regularization on ϕ_i for the i-th task, and therefore reduces the complexity of the solution space and leads to a better generalization.

To quantify the improvement of the generalization, let us denote the population meta-risk by

$$\mathcal{R} = \mathbb{E}_{\mathcal{T}_i\sim p(\mathcal{T})}\mathbb{E}_{(\mathbf{X}_i,\mathbf{Y}_i)\sim\mathcal{T}_i}[\mathcal{L}(f_{\phi_i}(\mathbf{X}_i),\mathbf{Y}_i)], \tag{11}$$

and the empirical version by

$$\begin{aligned}\mathcal{R}(\{\mathbf{Z}_i\}_{i=1}^{n_T}) =& \frac{1}{n_T}\sum_{i=1}^{n_T}\frac{1}{K^m}\sum_{j=1}^{K^m}\mathcal{L}(f_{\phi_i}(\mathbf{x}_{i,j}),\mathbf{y}_{i,j}) \\ =& \mathbb{E}_{\mathcal{T}_i\sim\hat{p}(\mathcal{T})}\mathbb{E}_{(\mathbf{X}_i,\mathbf{Y}_i)\sim\hat{p}(\mathcal{T}_i)}\mathcal{L}(f_{\phi_i}(\mathbf{X}_i),\mathbf{Y}_i).\end{aligned} \tag{12}$$

According to Theorem 1, we study the generalization problem by considering the following function class that is closely related to the dual problem of Eqn. (10)

$$\mathcal{F}_\mathcal{T} = \{\phi^\top\sigma(\mathbf{WX}) : \phi^\top\Sigma_{\sigma,\mathcal{T}}\phi \le \gamma\}, \tag{13}$$

where $\Sigma_{\sigma,\mathcal{T}} = \mathbb{E}_\mathcal{T}[\sigma(\mathbf{WX})\sigma(\mathbf{WX})^\top]$. Notation-wise, let us also define $\mu_{\sigma,\mathcal{T}} = \mathbb{E}_\mathcal{T}[\sigma(\mathbf{WX})]$. Further, we also assume the condition of the task distribution $\mathcal{T}$: for all $\mathcal{T}\sim p(\mathcal{T})$, $\mathcal{T}$ satisfies

$$rank(\Sigma_{\sigma,\mathcal{T}}) \le r, \|\Sigma_{\sigma,\mathcal{T}}^{\mathbf{W}\dagger/2}\mu_{\sigma,\mathcal{T}}\| \le B, \tag{14}$$

where $p(\mathcal{T})$ is the distribution of the task distribution.

We then have the following theorem showing the improvement on the meta-generalization gap induced by the MetaMix algorithm.

Theorem 1 *Suppose* $\mathbf{X}$, $\mathbf{Y}$ *and* ϕ *are all bounded, and also assume assumption Eqn. (14) holds. Then there exists constants* $C_1, C_2, C_3, C_4 > 0$*, such that for all* $f_\mathcal{T} \in \mathcal{F}_\mathcal{T}$*, we have, with probability at least* $1-\delta$*,*

$$\begin{aligned}|\mathcal{R}(\{\mathbf{Z}_i\}_{i=1}^{n_T}) - \mathcal{R}| \le& C_1\sqrt{\frac{\gamma\cdot(r+B)}{K^m}} + C_2\sqrt{\frac{\log(n_T/\delta)}{K^m}} \\ &+C_3\sqrt{\frac{\gamma\cdot B+1}{n_T}} + C_4\sqrt{\frac{\log(1/\delta)}{n_T}}.\end{aligned} \tag{15}$$

According to Lemma 1, Mixup is regularizing $\phi^\top\Sigma_{\sigma,T}\phi$ and making γ small. With this interpretation, Theorem 2 then suggests that a smaller value of γ induced by Mixup will help reduce the generalization error, and therefore mitigate the overfitting.

Analysis of Channel Shuffle. We then analyze the channel shuffle strategy under the same two-layer neural network model considered above, with binary class $\mathbf{y}_{i,j} \in \{0,1\}$. Instead of applying the mixup on $\mathbf{Z}_i = \{\mathbf{x}_{i,j},\mathbf{y}_{i,j}\}_{j=1}^{K^m} := \{\mathbf{x}_{i,j;0},0\}_{j=1}^{K^{m_0}} \cup \{\mathbf{x}_{i,j;1},1\}_{j=1}^{K^{m_1}}$, we now apply the channel shuffle strategy. Specifically, we consider the shuffled data $\mathbf{Z}_i^{cf} = \{\mathbf{x}_{i,j}^{cf},\mathbf{y}_{i,j}\}_{j=1}^{K^m} = \{\mathbf{x}_{i,j;0}^{cf},0\}_{j=1}^{K^{m_0}} \cup \{\mathbf{x}_{i,j;1}^{cf},1\}_{j=1}^{K^{m_1}}$. According to Eqn. (6), $\{\mathbf{x}_{i,j;k}^{cf}\}$ ($k\in\{0,1\}$) is constructed as

$$\begin{aligned}\mathbf{x}_{i,j;k}^{cf} = \frac{1}{\delta}\cdot(\mathbf{R}\sigma(\mathbf{W}\mathbf{x}_{i,j;k}) + (\mathbf{I}-\mathbf{R})\sigma(\mathbf{W}\mathbf{x}_{i,j';1-k})) \\ \text{for } j\in[K^{m_k}], k\in\{0,1\}.\end{aligned} \tag{16}$$

Let us denote such randomness by ξ. Recall that $\mathbf{R} = \text{diag}(r_1,...,r_p)$ with $r_t \sim \text{Bernoulli}(\delta)$, the scaling $\frac{1}{\delta}$ is added for technical convenience. Since the last layer is linear, the scaling $\frac{1}{\delta}$ will not affect the training and prediction results.

We now define $\mathcal{L}(\mathbf{Z}_i^{cf}) = \frac{1}{K^m}\sum_{j=1}^{K^m}\mathcal{L}(\phi_i{}^\top(\mathbf{x}_{i,j}^{cf}),\mathbf{y}_{i,j})$. For a generic vector $\mathbf{v}\in\mathbb{R}^p$, we denote $\mathbf{v}^{\circ 2} = (v_1^2,...,v_p^2)$ and $diag(\mathbf{v}^{\circ 2}) = diag(v_1^2,...,v_p^2)$ as the diagonal matrix with diagonal elements $(v_1^2,...,v_p^2)$. We then have the following theorem on the second-order approximation of $\mathcal{L}(\mathbf{Z}_i^{cf})$.

Theorem 2 *Consider the model set-up described above and recall that* ξ *is the randomness involved in the data argumentation. Assume the training data is preprocessed as* $\frac{1}{K^{m_0}}\sum_{j=1}^{K^{m_0}}\sigma(\mathbf{W}\mathbf{x}_{i,j;0}) = \frac{1}{K^{m_1}}\sum_{j=1}^{K^{m_1}}\sigma(\mathbf{W}\mathbf{x}_{i,j;1}) = 0$*. There exists a constant* $c>0$*, such that the second-order approximation of* $\mathbb{E}_\xi\mathcal{L}(\mathbf{Z}_i^{cf})$ *is given by*

$$\begin{aligned}&\mathcal{L}(\mathbf{Z}_i) + \frac{1-\delta}{\delta}\phi_i^\top(\frac{1}{K^m}\sum_{j=1}^{K^m}diag(\sigma(\mathbf{W}\mathbf{x}_{i,j})^{\circ 2})\phi_i + \\ &\quad+\frac{1-\delta}{\delta}\phi_i^\top(\frac{1}{K^{m_0}}\sum_{j=1}^{K^{m_0}}\sigma(\mathbf{W}\mathbf{x}_{i,j;0})\sigma(\mathbf{W}\mathbf{x}_{i,j;0})^\top \\ &\quad+\frac{1}{K^{m_1}}\sum_{j=1}^{K^{m_1}}\sigma(\mathbf{W}\mathbf{x}_{i,j;1})\sigma(\mathbf{W}\mathbf{x}_{i,j;1})^\top)\phi_i.\end{aligned} \tag{17}$$

Theorem 2 suggests that the Channel Shuffle algorithm will also impose a quadratic data-adaptive regularization on ϕ_i, and the second quadratic term resembles the one induced by MetaMix in Lemma 1. As a result, it will make the γ in Theorem 2 smaller and further reduce the overfitting. We provide more details and the full proof about theoretical analysis in Appendix C.

5. Discussion with Related Works

One influential line of meta-learning algorithms is learning a transferable metric space between samples from previous tasks (Mishra et al., 2018; Oreshkin et al., 2018; Snell et al., 2017; Vinyals et al., 2016), which classify samples via lazy learning with the learned distance metric (e.g., Euclidean distance (Snell et al., 2017), cosine distance (Vinyals et al., 2016)). However, their applications are limited to classification problems, being infeasible in other problems (e.g.,

regression). In this work, we focus on gradient-based meta-learning algorithms that learn a well-generalized model initialization from meta-training tasks (Finn & Levine, 2018; Finn et al., 2017; 2018; Flennerhag et al., 2020; Grant et al., 2018; Lee & Choi, 2018; Li et al., 2017; Park & Oliva, 2019), being agnostic to problems. This initialization is adapted to each task via the support set, and in turn the initialization is updated by maximizing the generalization performance on the query set. These approaches are at high risk of overfitting the meta-training tasks and generalizing poorly to meta-testing tasks.

Common techniques increase the generalization capability via regularizations such as weight decay (Krogh & Hertz, 1992), dropout (Gal & Ghahramani, 2016; Srivastava et al., 2014), and incorporating noise (Achille & Soatto, 2018; Alemi et al., 2017; Tishby & Zaslavsky, 2015). However, the adapted model by only a few steps on the support set in the inner-loop likely performs poorly on the query set. To improve such generalization for better adaptation, either the number of parameters to adapt is reduced (Raghu et al., 2020; Zintgraf et al., 2019; Oh et al., 2021) or adpative noise is added (Lee et al., 2020). The contribution of addressing this inner-loop overfitting towards meta-regularization, though positive, is limited.

Until very recently, two regularizers were proposed to specifically improve meta-generalization, including MR-MAML (Yin et al., 2020) which regularizes the search space of the initialization while meanwhile allows it to be sufficiently adapted in the inner-loop, and TAML (Jamal & Qi, 2019) enforcing the initialization to behave similarly across tasks. Instead of imposing regularizers on the initialization, Rajendran et al. (2020) proposed to inject a random constant noise to labels of both support and query sets. The shared noise, however, is easy to be learned in the inner-loop. Besides, as we prove in Appendix A.3, this augmentation fails to meet the second criterion in Definition 1 and therefore little additional information is provided to meta-train the initialization. Our work takes sufficiently powerful ways actively soliciting more data to meta-train the initialization. Note that our task augmentation strategies are more than just a simple application of conventional data augmentation strategies (Cubuk et al., 2019; Verma et al., 2019; Zhang et al., 2018), which have been proved in both (Lee et al., 2020) and our experiments to have a very limited role. We initiate to include more query data that satisfy the proposed Criterion 1 in the meta-training phase, so that the dependence on support sets during inner-loop adaptation is increased and the meta-generalization is improved.

6. Experiments

To show the effectiveness of MetaMix, we conduct experiments on three meta-learning problems, namely: (1) drug activity prediction, (2) pose prediction, and (3) image classification. We apply MetaMix on four gradient-based meta-learning algorithms, including MAML (Finn et al., 2017), MetaSGD (Li et al., 2017), T-Net (Lee & Choi, 2018), and ANIL (Raghu et al., 2020). For comparison, we consider the following regularizers: Weight Decay as the traditional regularizer, CAVIA (Zintgraf et al., 2019) and Meta-dropout (Lee et al., 2020) which regularize the inner-loop, and MR-MAML (Yin et al., 2020), TAML (Jamal & Qi, 2019), and Meta-Aug (Rajendran et al., 2020), all of which handle meta-generalization.

6.1. Drug Activity Prediction

Experimental Setup. We solve a real-world application of drug activity prediction (Martin et al., 2019) where there are 4,276 target assays (i.e., tasks) each of which consists of a few drug compounds with tested activities against the target protein. We randomly selected 100 assays for meta-testing, 76 for meta-validation and the rest for meta-training. We repeat the random process four times and construct four groups of meta-testing assays for evaluation. Following (Martin et al., 2019), we evaluate the square of Pearson coefficient R^2 between the predicted $\hat{\mathbf{y}}_i^q$ and the groundtruth $\mathbf{y}_i^q$ of all query samples for each i-th task, and report the mean and median R^2 values over all meta-testing assays as well as the number of assays with $R^2 > 0.3$ which is deemed as an indicator of reliability in pharmacology. We use a base model of two fully connected layers with 500 hidden units. In $\mathrm{Beta}(\alpha, \beta)$, we set $\alpha = \beta = 0.5$. More details on the dataset and settings are discussed in Appendix D.1.

Performance. In practice, we notice that only updating the final layer in the inner-loop achieves the best performance, which is equivalent to ANIL. Thus, we apply this inner-loop update strategy to all baselines. For stability, here we also use ANIL++ (Antoniou et al., 2019) which stabilizes ANIL for comparison. In Table 1, we compare MetaMix with the baselines on the four drug evaluation groups. We observe that MetaMix consistently improves the performance despite of the backbone meta-learning algorithms (i.e., ANIL, ANIL++, MetaSGD, T-Net) in all scenarios. In addition, ANIL-MetaMix outperforms other anti-overfitting strategies. Particularly, compared to Meta-Aug, the better performance of ANIL-MetaMix indicates that additional information provided by MetaMix benefits the meta-generalization. In summary, the consistent superior performance, even significantly better than the state-of-the-art pQSAR-max for this dataset, demonstrates that (1) MetaMix is compatible with existing meta-learning algorithms; (2) MetaMix is capable of improving the meta-generalization ability. Besides, in Appendix E.1, we investigate the influence of different hyperparameter settings (e.g., α in $\mathrm{Beta}(\alpha, \alpha)$), and demonstrate the robustness of MetaMix under different settings.

Table 1. Performance of drug activity prediction.

Model	Group 1			Group 2			Group 3			Group 4		
	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3
pQSAR-max (Martin et al., 2019)	0.390	0.335	51	0.335	0.280	44	0.373	0.315	50	0.362	0.260	46
Weight Decay	0.307	0.228	40	0.243	0.157	34	0.259	0.171	38	0.290	0.241	47
CAVIA	0.300	0.232	42	0.234	0.132	35	0.260	0.184	39	0.317	0.292	46
Meta-dropout	0.319	0.203	41	0.250	0.172	35	0.281	0.214	39	0.316	0.275	47
Meta-Aug	0.317	0.201	43	0.253	0.193	38	0.286	0.220	41	0.303	0.224	42
MR-ANIL	0.297	0.202	41	0.232	0.152	32	0.289	0.217	40	0.293	0.249	43
TAML	0.296	0.200	41	0.260	0.203	36	0.260	0.227	40	0.308	0.281	46
MetaSGD	0.331	0.224	45	0.249	0.187	33	0.282	0.226	40	0.312	0.287	48
T-Net	0.323	0.264	46	0.236	0.170	29	0.285	0.220	43	0.285	0.239	42
ANIL	0.299	0.184	41	0.226	0.143	30	0.268	0.199	37	0.304	0.282	48
ANIL++	0.367	0.299	50	0.315	0.252	43	0.335	0.289	48	0.362	0.324	51
MetaSGD-MetaMix	0.364	0.296	49	0.271	0.230	45	0.312	0.267	48	0.338	0.319	51
T-Net-MetaMix	0.352	0.291	50	0.276	0.229	42	0.310	0.285	47	0.336	0.298	50
ANIL-MetaMix	0.347	0.292	49	0.301	0.282	47	0.302	0.258	45	0.348	0.303	51
ANIL++-MetaMix	**0.413**	**0.393**	**59**	**0.337**	**0.301**	**51**	**0.381**	**0.362**	**55**	**0.380**	**0.348**	**55**

Analysis of Overfitting. In Figure 2, we visualize the meta-training and meta-testing performance of ANIL, ANIL-MetaMix and other two representative anti-overfitting strategies (i.e., MR-ANIL, Meta-Aug) with respect to the training iteration. Interestingly, we find (1) in the meta-testing phase, applying MetaMix significantly increases the performance gap between pre-update (θ_0) and post-update (ϕ_i), indicating that MetaMix improves the dependence of target prediction on support sets, and therefore alleviates memorization overfitting; (2) compared to Meta-Aug and MR-ANIL, the worse pre-update meta-training performance but better post-update meta-testing performance of MetaMix demonstrates its superiority to mitigate the learner overfitting.

Effect of Data Mixture Strategy in MetaMix. To further investigate where the improvement stems from, we adopt five different mixup strategies for meta-training. The results are reported in Table 2. We use Mixup($\mathcal{D}^m$, $\mathcal{D}^n$) to denote the mixup of data $\mathcal{D}^m$ and $\mathcal{D}^n$ (e.g., Mixup($\mathcal{D}^s$, $\mathcal{D}^q$) in our case). $\mathcal{D}^{cob} = \mathcal{D}^s \oplus \mathcal{D}^q$ represents the concatenation of $\mathcal{D}^s$ and $\mathcal{D}^q$. In drug activity prediction, since the support and query sets are pre-split based on the biological domain knowledge, we also introduce set shuffle as another ablation model by randomly shuffling the pre-split sets. In Table 2, we find that (1) MetaMix achieves the best performance compared with other ablation models; (2) the fact that MetaMix enjoys better performance than Mixup($\mathcal{D}^q, \mathcal{D}^q$) suggests that MetaMix is much more than simple data augmentation – it increases the dependency of the learner on support sets and thereby minimizes the memorization; (3) involving the support set only is insufficient for meta-generalization due to its relative small gradient norm, which is further verified by the unsatisfactory performance of $\mathcal{D}^s \oplus \mathcal{D}^q$ compared with MetaMix .

Analysis of Criteria. We further analyze augmentation methods on drug data (Group 1) with respect to the two criteria (C1, C2) we propose and the CE-increasing criterion H(Y|X)↑ proposed by Meta-Aug. We report the results in Table 3, where Mix-all applies Mixup to the whole dataset without differentiating different tasks. We observe that C1 and C2 are qualified to guide the design of task augmentation methods, as evidenced in Table 3 where methods satisfying more of C1 and C2 tend to perform better.

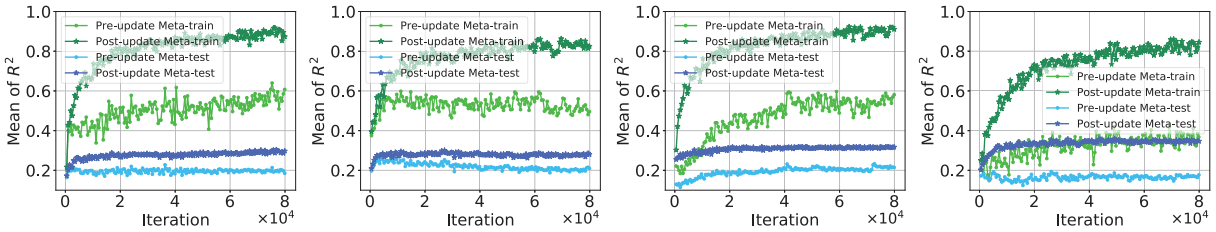
6.2. Pose Prediction

Experimental Setup. Following (Yin et al., 2020), we use the regression dataset created from Pascal 3D data (Xiang et al., 2014), where a 128×128 grey-scale image is used as input and the orientation relative to a fixed pose labels each image. 50 and 15 objects are randomly selected for meta-training and meta-testing, respectively. Following (Yin et al., 2020), the base model consists of an encoder with three convolutional blocks and a decoder with four convolutional blocks. For MetaMix, we set $\alpha = \beta = 0.5$ in $\mathrm{Beta}(\alpha, \beta)$ and only perform Mainfold Mixup on the decoder (see Appendix D.2 for detailed settings).

Results. Table 4 shows the performance (averaged MSE with 95% confidence interval) of baselines and MetaMix under 10/15-shot scenarios. The inner-loop regularizers are not as effective as MR-MAML, TAML and Meta-Aug in improving meta-generalization; MAML-MetaMix and Meta-Aug significantly improve MR-MAML, suggesting the effectiveness of bringing more data in than imposing meta-regularizers only. The better performance of MAML-MetaMix than Meta-Aug further verifies the effectiveness of introducing additional knowledge to learn the initialization. We also investigate the influence of mixup strategies and hyperparameters on pose prediction in Appendix F.1 and F.2, respectively. The results again advocate the effectiveness

Table 2. Effect of mixture strategies on drug activity prediction. All strategies are applied on ANIL++.

Strategies	Group 1			Group 2			Group 3			Group 4		
	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3	Mean	Med.	>0.3
$\mathcal{D}^q$	0.367	0.299	50	0.315	0.252	43	0.335	0.289	48	0.362	0.324	51
Set Shuffle	0.371	0.352	55	0.293	0.224	42	0.339	0.297	50	0.360	0.300	50
Mixup($\mathcal{D}^s$, $\mathcal{D}^s$)	0.224	0.164	33	0.210	0.164	31	0.214	0.154	29	0.191	0.141	22
Mixup($\mathcal{D}^q$, $\mathcal{D}^q$)	0.388	0.354	55	0.322	0.264	46	0.341	0.306	50	0.358	0.325	53
$\mathcal{D}^{cob} = \mathcal{D}^s \oplus \mathcal{D}^q$	0.376	0.324	52	0.301	0.242	44	0.333	0.329	51	0.336	0.281	48
MetaMix	**0.413**	**0.393**	**59**	**0.337**	**0.301**	**51**	**0.381**	**0.362**	**55**	**0.380**	**0.348**	**55**



(a) : ANIL (b) : MR-ANIL (c) : Meta-Aug (d) : ANIL-MetaMix

Figure 2. Overfitting analysis on Group 1 of drug activity prediction. All models use the same inner-loop update strategy as ANIL.

Table 3. Criteria analysis on Group 1 of drug activity prediction. All models use ANIL as the backbone meta-learning algorithm.

Aug. Method	C1	C2	H(Y|X)↑	Mean R^2
Mix-All				0.292
Mixup($\mathcal{D}^q$, $\mathcal{D}^q$)		√	√	0.322
Meta-Aug	√		√	0.317
ANIL-MetaMix	√	√	√	**0.347**

Table 4. Performance (MSE ± 95% confidence interval) of pose prediction.

Model	10-shot	15-shot
Weight Decay	2.772 ± 0.259	2.307 ± 0.226
CAVIA	3.021 ± 0.248	2.397 ± 0.191
Meta-dropout	3.236 ± 0.257	2.425 ± 0.209
Meta-Aug	2.553 ± 0.265	2.152 ± 0.227
MR-MAML	2.907 ± 0.255	2.276 ± 0.169
TAML	2.785 ± 0.261	2.196 ± 0.163
ANIL	6.746 ± 0.416	6.513 ± 0.384
MAML	3.098 ± 0.242	2.413 ± 0.177
MetaSGD	2.803 ± 0.239	2.331 ± 0.182
T-Net	2.835 ± 0.189	2.609 ± 0.213
ANIL-MetaMix	6.354 ± 0.393	6.112 ± 0.381
MAML-MetaMix	2.438 ± 0.196	2.003 ± 0.147
MetaSGD-MetaMix	$\mathbf{2.390 \pm 0.191}$	$\mathbf{1.952 \pm 0.154}$
T-Net-MetaMix	2.563 ± 0.201	2.418 ± 0.182

and robustness of the proposed mixup strategy in improving the meta-generalization ability.

6.3. Image Classification

Experimental Setup. For image classification problems, standard benchmarks (e.g., Omniglot (Lake et al., 2011) and MiniImagenet (Vinyals et al., 2016)) are considered as mutually-exclusive tasks by introducing the shuffling mechanism of labels, which significantly alleviates the meta-overfitting issue (Yin et al., 2020). To show the power of proposed augmentation strategies, following (Yin et al., 2020), we adopt the non-mutually-exclusive setting for each image classification benchmark: each class with its classification label remains unchanged across different meta-training tasks during meta-training. Besides, we study image classification for heterogeneous tasks in Appendix G.1. We use the multi-dataset in (Yao et al., 2019) which consists of four subdatasets, i.e., Bird, Texture, Aircraft, and Fungi. The non-mutually-exclusive setting is also applied to this multi-dataset. Three representative heterogeneous meta-learning algorithms (i.e., MMAML (Vuorio et al., 2019), HSML (Yao et al., 2019), ARML (Yao et al., 2020)) are taken as baselines and applied with task augmentation stategies. For each task, the classical N-way, K-shot setting is used to evaluate the performance. We use the standard four-block convolutional neural network as the base model. We set $\alpha = \beta = 2.0$ for all datasets. Detailed descriptions of experiment settings and hyperparameters are discussed in Appendix D.3.

Results. In Table 5 and Appendix G.1, we report the performance (accuracy with 95% confidence interval) on homogeneous datasets (i.e., Omniglot, MiniImagenet) and heterogeneous datasets, respectively. As described in Section 3, we will use Channel Shuffle enhanced MetaMix (MMCF) in image classification problems. Aligned with other problems, in all non-mutually-exclusive datasets, applying the MMCF consistently improves existing meta-learning algorithms. For example, MAML-MMCF significantly boosts MAML and most importantly outperforms MR-MAML, substanti-

Table 5. Performance (accuracy $\pm$ 95% confidence interval) of image classification on Omniglot and MiniImagenet.

Model	Omniglot		MiniImagenet	
	20-way 1-shot	20-way 5-shot	5-way 1-shot	5-way 5-shot
Weight Decay	$86.81 \pm 0.64\%$	$96.20 \pm 0.17\%$	$33.19 \pm 1.76\%$	$52.27 \pm 0.96\%$
CAVIA	$87.63 \pm 0.58\%$	$94.16 \pm 0.20\%$	$34.27 \pm 1.79\%$	$50.23 \pm 0.98\%$
MR-MAML	$89.28 \pm 0.59\%$	$96.66 \pm 0.18\%$	$35.00 \pm 1.60\%$	$54.39 \pm 0.97\%$
Meta-dropout	$85.60 \pm 0.63\%$	$95.56 \pm 0.17\%$	$34.32 \pm 1.78\%$	$52.40 \pm 0.96\%$
TAML	$87.50 \pm 0.63\%$	$95.78 \pm 0.19\%$	$33.16 \pm 1.68\%$	$52.78 \pm 0.97\%$
MAML	$87.40 \pm 0.59\%$	$93.51 \pm 0.25\%$	$32.93 \pm 1.70\%$	$51.95 \pm 0.97\%$
MetaSGD	$87.72 \pm 0.61\%$	$95.52 \pm 0.18\%$	$33.70 \pm 1.63\%$	$52.14 \pm 0.92\%$
T-Net	$87.71 \pm 0.62\%$	$95.67 \pm 0.20\%$	$33.73 \pm 1.72\%$	$54.04 \pm 0.99\%$
ANIL	$88.35 \pm 0.56\%$	$95.85 \pm 0.19\%$	$34.13 \pm 1.67\%$	$52.59 \pm 0.96\%$
MAML-MMCF	$92.06 \pm 0.51\%$	$97.95 \pm 0.17\%$	$39.26 \pm 1.79\%$	$58.96 \pm 0.95\%$
MetaSGD-MMCF	$\mathbf{93.59 \pm 0.45\%}$	$98.24 \pm 0.16\%$	$\mathbf{40.06 \pm 1.76\%}$	$\mathbf{60.19 \pm 0.96\%}$
T-Net-MMCF	$93.27 \pm 0.46\%$	$98.09 \pm 0.15\%$	$38.33 \pm 1.73\%$	$59.13 \pm 0.99\%$
ANIL-MMCF	$92.24 \pm 0.48\%$	$\mathbf{98.36 \pm 0.13\%}$	$37.94 \pm 1.75\%$	$59.03 \pm 0.93\%$

Table 6. Performance (accuracy $\pm$ 95% confidence interval) of MiniImagenet and Omniglot w.r.t. different data augmentation strategies applied on MAML.

Strategy	Omniglot		MiniImagenet	
	20-way 1-shot	20-way 5-shot	5-way 1-shot	5-way 5-shot
$\mathcal{D}^q$	$87.40 \pm 0.59\%$	$93.51 \pm 0.25\%$	$32.93 \pm 1.70\%$	$51.95 \pm 0.97\%$
Mixup($\mathcal{D}^s$, $\mathcal{D}^s$)	$46.98 \pm 0.92\%$	$85.56 \pm 0.28\%$	$24.39 \pm 1.48\%$	$33.18 \pm 0.82\%$
Mixup($\mathcal{D}^q$, $\mathcal{D}^q$)	$90.65 \pm 0.56\%$	$96.90 \pm 0.16\%$	$34.56 \pm 1.77\%$	$55.80 \pm 0.97\%$
$\mathcal{D}^{cob} = \mathcal{D}^s \oplus \mathcal{D}^q$	$86.74 \pm 0.54\%$	$95.54 \pm 0.19\%$	$33.33 \pm 1.70\%$	$51.97 \pm 0.96\%$
MetaMix	$91.53 \pm 0.53\%$	$97.63 \pm 0.15\%$	$38.53 \pm 1.79\%$	$57.55 \pm 1.01\%$
Channel Shuffle	$89.81 \pm 0.55\%$	$97.10 \pm 0.17\%$	$35.50 \pm 1.73\%$	$54.52 \pm 0.96\%$
MMCF	$\mathbf{92.06 \pm 0.51\%}$	$\mathbf{97.95 \pm 0.17\%}$	$\mathbf{39.26 \pm 1.79\%}$	$\mathbf{58.96 \pm 0.95\%}$

ating the effectiveness of MMCF in improving the meta-generalization ability. It is worth mentioning that we also conduct the experiments on the standard mutually-exclusive setting of MiniImagenet in Appendix G.2. Though the label shuffling has significantly mitigated meta-overfitting, applying MMCF still improves the meta-generalization to some extent. Besides, under the MiniImagenet 5-shot scenario, we investigate the influence of different hyperparameters, including sampling λ from the Beta distribution with different values of α and β, varying different fixed values of λ, and adjusting the layer to mixup (i.e., $\mathcal{C}$ in Eqn. (4)) in Appendix G.3. All these studies indicate the robustness of MetaMix and Channel Shuffle in improving the meta-generalization.

Ablation Study. To align with other problems, for MMCF, we vary the mixup and data augmentation strategies (i.e., MetaMix, Channel Shuffle) in image classification in Table 6. First, comparing MetaMix to other data mixup strategies, we again corroborate the effectiveness of MetaMix in improving meta-generalization. Second, we compare MMCF with MetaMix and Channel Shuffle, the better performance of MMCF indicates the additional effects of Channel Shuffle to enhance MetaMix in classification problems, as our theoretic analyses suggest.

7. Conclusion

Current gradient-based meta-learning algorithms are at high risk of overfitting on meta-training tasks but poorly generalizing to meta-testing tasks. To address this issue, we propose two novel data augmentation strategies – MetaMix and Channel Shuffle, which actively involve more data in the outer-loop optimization process. Specifically, MetaMix linearly interpolates the features and labels of support and target sets. In classification problems, MetaMix is further enhanced by Channel Shuffle, which randomly replaces a subset of channels with the corresponding ones from another class. We theoretically demonstrate that all strategies can improve the meta-generalization capability. The state-of-the-art results on different real-world datasets demonstrate the effectiveness and compatibility of the proposed methods.

Acknowledgments

H.Y. and Z.L. are supported in part by NSF awards IIS-#1652525 and IIS-#1618448. L.Z. is supported by NSF awards DMS-#2015378. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

References

Achille, A. and Soatto, S. Information dropout: Learning optimal representations through noisy computation. *T-PAMI*, 40(12):2897–2905, 2018.

Alemi, A. A., Fischer, I., Dillon, J. V., and Murphy, K. Deep variational information bottleneck. In *ICLR*, 2017.

Antoniou, A., Edwards, H., and Storkey, A. How to train your maml. In *ICLR*, 2019.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In *CVPR*, pp. 113–123, 2019.

Dou, Z.-Y., Yu, K., and Anastasopoulos, A. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *EMNLP*, 2019.

Finn, C. and Levine, S. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. *ICLR*, 2018.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pp. 1126–1135, 2017.

Finn, C., Xu, K., and Levine, S. Probabilistic model-agnostic meta-learning. *NeurIPS*, 2018.

Flennerhag, S., Rusu, A. A., Pascanu, R., Yin, H., and Hadsell, R. Meta-learning with warped gradient descent. In *ICLR*, 2020.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, pp. 1050–1059, 2016.

Grant, E., Finn, C., Levine, S., Darrell, T., and Griffiths, T. Recasting gradient-based meta-learning as hierarchical bayes. In *ICLR*, 2018.

Gu, J., Wang, Y., Chen, Y., Cho, K., and Li, V. O. Meta-learning for low-resource neural machine translation. In *EMNLP*, 2018.

Jamal, M. A. and Qi, G.-J. Task agnostic meta-learning for few-shot learning. In *CVPR*, pp. 11719–11727, 2019.

Kang, B., Liu, Z., Wang, X., Yu, F., Feng, J., and Darrell, T. Few-shot object detection via feature reweighting. In *ICCV*, pp. 8420–8429, 2019.

Krogh, A. and Hertz, J. A. A simple weight decay can improve generalization. In *NeurIPS*, pp. 950–957, 1992.

Lake, B., Salakhutdinov, R., Gross, J., and Tenenbaum, J. One shot learning of simple visual concepts. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.

Lee, H. B., Nam, T., Yang, E., and Hwang, S. J. Meta dropout: Learning to perturb latent features for generalization. In *ICLR*, 2020.

Lee, Y. and Choi, S. Gradient-based meta-learning with learned layerwise metric and subspace. In *ICML*, pp. 2933–2942, 2018.

Li, Z., Zhou, F., Chen, F., and Li, H. Meta-sgd: Learning to learn quickly for few shot learning. *arXiv preprint arXiv:1707.09835*, 2017.

Liu, M.-Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., and Kautz, J. Few-shot unsupervised image-to-image translation. In *ICCV*, pp. 10551–10560, 2019.

Madotto, A., Lin, Z., Wu, C.-S., and Fung, P. Personalizing dialogue agents via meta-learning. In *ACL*, 2019.

Martin, E. J., Polyakov, V. R., Zhu, X.-W., Tian, L., Mukherjee, P., and Liu, X. All-assay-max2 pqsar: Activity predictions as accurate as four-concentration ic50s for 8558 novartis assays. *Journal of chemical information and modeling*, 59(10):4450–4459, 2019.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *ICLR*, 2018.

Oh, J., Yoo, H., Kim, C., and Yun, S.-Y. Boil: Towards representation change for few-shot learning. In *ICLR*, 2021.

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

Park, E. and Oliva, J. B. Meta-curvature. In *NeurIPS*, pp. 3309–3319, 2019.

Raghu, A., Raghu, M., Bengio, S., and Vinyals, O. Rapid learning or feature reuse? towards understanding the effectiveness of maml. In *ICLR*, 2020.

Rajendran, J., Irpan, A., and Jang, E. Meta-learning requires meta-augmentation. *NeurIPS*, 2020.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *NIPS*, pp. 4077–4087, 2017.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. Learning to compare: Relation network for few-shot learning. In *CVPR*, pp. 1199–1208, 2018.

Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 1998.

Tishby, N. and Zaslavsky, N. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pp. 1–5. IEEE, 2015.

Verma, V., Lamb, A., Beckham, C., Najafi, A., Mitliagkas, I., Courville, A., Lopez-Paz, D., and Bengio, Y. Manifold mixup: Better representations by interpolating hidden states. *ICML*, 2019.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *NIPS*, pp. 3630–3638, 2016.

Vuorio, R., Sun, S.-H., Hu, H., and Lim, J. J. Multimodal model-agnostic meta-learning via task-aware modulation. In *NeurIPS*, pp. 1–12, 2019.

Xiang, Y., Mottaghi, R., and Savarese, S. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, pp. 75–82. IEEE, 2014.

Xie, A., Singh, A., Levine, S., and Finn, C. Few-shot goal inference for visuomotor learning and planning. In *CoRL*, pp. 40–52, 2018.

Yao, H., Wei, Y., Huang, J., and Li, Z. Hierarchically structured meta-learning. In *ICML*, 2019.

Yao, H., Wu, X., Tao, Z., Li, Y., Ding, B., Li, R., and Li, Z. Automated relational meta-learning. In *ICLR*, 2020.

Yin, M., Tucker, G., Zhou, M., Levine, S., and Finn, C. Meta-learning without memorization. *ICLR*, 2020.

Yu, T., Finn, C., Xie, A., Dasari, S., Zhang, T., Abbeel, P., and Levine, S. One-shot imitation from observing humans via domain-adaptive meta-learning. In *RSS*, 2018.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

Zhang, L., Deng, Z., Kawaguchi, K., Ghorbani, A., and Zou, J. How does mixup help with robustness and generalization? In *ICLR*, 2021.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. Random erasing data augmentation. In *AAAI*, 2020.

Zintgraf, L. M., Shiarlis, K., Kurin, V., Hofmann, K., and Whiteson, S. Fast context adaptation via meta-learning. In *ICML*, 2019.