

---

# Hierarchically Structured Meta-learning

---

Huaxiu Yao<sup>† 1</sup> Ying Wei<sup>2</sup> Junzhou Huang<sup>2</sup> Zhenhui Li<sup>1</sup>

## Abstract

In order to learn quickly with few samples, meta-learning utilizes prior knowledge learned from previous tasks. However, a critical challenge in meta-learning is task uncertainty and heterogeneity, which can not be handled via globally sharing knowledge among tasks. In this paper, based on gradient-based meta-learning, we propose a hierarchically structured meta-learning (HSML) algorithm that explicitly tailors the transferable knowledge to different clusters of tasks. Inspired by the way human beings organize knowledge, we resort to a hierarchical task clustering structure to cluster tasks. As a result, the proposed approach not only addresses the challenge via the knowledge customization to different clusters of tasks, but also preserves knowledge generalization among a cluster of similar tasks. To tackle the changing of task relationship, in addition, we extend the hierarchical structure to a continual learning environment. The experimental results show that our approach can achieve state-of-the-art performance in both toy-regression and few-shot image classification problems.

## 1. Introduction

Learning quickly with a few samples is one of the key characteristics of human intelligence, while it remains a daunting challenge for artificial intelligence. Learning to learn (a.k.a., meta-learning) (Braun et al., 2010), as a common practice to address this challenge, leverages the transferable knowledge learned from previous tasks to improve the learning effectiveness in a new task. There have been several lines of meta-learning algorithms, including recurrent network based methods (Ravi & Larochelle, 2016), optimizer based

methods (Andrychowicz et al., 2016), nearest neighbours based methods (Snell et al., 2017; Vinyals et al., 2016) and gradient descent based methods (Finn et al., 2017), which instantiate the transferable knowledge as latent representations, an optimizer, a metric space, and parameter initialization, respectively.

Despite their early success in few-shot image classification (Ravi & Larochelle, 2016) and machine translation (Gu et al., 2018), most of the existing meta-learning algorithms assume the transferable knowledge to be globally shared across all tasks. As a consequence, they suffer from handling a sequence of tasks originated from different distributions. At the other end of the spectrum, recently, a few research works (Finn et al., 2018; Yoon et al., 2018a; Lee & Choi, 2018) try to fix the problem by tailoring the transferable knowledge to each task. Yet the downside of such methods lies in the impaired knowledge generalization among closely correlated tasks (e.g., the tasks sampled from the same distribution).

Hence we are motivated to pursue a meta-learning framework to effectively balance generalization and customization. The inspiration comes from a hypothesis which has been formulated and tested by psychological researchers (Gershman et al., 2010; 2014). The hypothesis suggests that the key to human beings' capability of solving a task with little training data is the way how human beings organize the learned knowledge from tasks. As bits of tasks impinge on us, we human beings cluster the tasks into several states based on task similarity, so that the learning occurs within each cluster instead of across cluster boundaries. Thus, when a new task arrives, it can either quickly take advantage of the knowledge learned within the cluster it belongs to or initiate a new cluster if it is wildly different from any existing clusters.

Inspired by this, we propose a novel meta-learning framework called **H**ierarchically **S**tructured **M**eta-**L**earning (HSML). The key idea of the HSML is to enhance the meta-learning effectiveness by promoting knowledge customization to different clusters of tasks but simultaneously preserving knowledge generalization among a cluster of closely related tasks. In this paper, without loss of generality, we ground HSML on a gradient based meta learning algorithm (Finn et al., 2017) with the transferable knowl-

<sup>†</sup>Part of the work was done when the author interned in Tencent AI Lab. <sup>1</sup>College of Information Science and Technology, Pennsylvania State University, PA, USA <sup>2</sup>Tencent AI Lab, Shenzhen, China. Correspondence to: Ying Wei <judyweiying@gmail.com>.

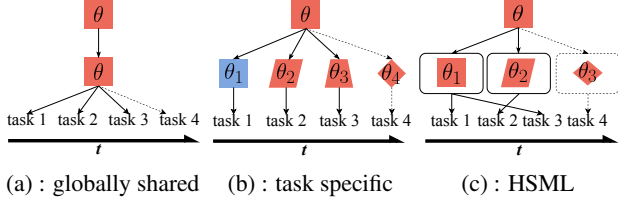


Figure 1. Pictorial illustration of the difference between the proposed HSML and the other two representative lines of gradient based meta-learning algorithms.

edge instantiated as parameter initializations. Specifically, first, the HSML resorts to a hierarchical clustering structure to perform soft clustering on tasks. The representation of each task is learned from either of the two proposed candidate aggregators, i.e., pooling autoencoder aggregator and recurrent autoencoder aggregator, and is passed to the hierarchical clustering structure to obtain the clustering result of this task. The sequentially incoming tasks, in turn, update the clustering structure. Especially, if the existing structure does not fit the task, we dynamically expand the structure. Secondly, a globally shared parameter initialization is tailored to each cluster via a parameter gate, to serve as the initializations for all tasks belonging to the cluster.

Again we would highlight the contribution of the proposed HSML: 1) it achieves a better balance between generalization and customization of the transferable knowledge, so that it empirically outperforms state-of-the-art meta-learning algorithms in both toy regression and few-shot image classification problems; 2) it is interpretable in terms of the task relationship; 3) it has been theoretically proved to be superior than existing gradient-based meta-learning algorithms.

## 2. Related Work

Meta-learning, allowing machines to learn new skills or adapt to new environments rapidly with a few training examples, has demonstrated success in both supervised learning such as few-shot image classification and reinforcement learning settings. There are four common approaches: 1) use a recurrent neural network equipped with either external or internal memory storing and querying meta-knowledge (Munkhdalai & Yu, 2017; Santoro et al., 2016; Munkhdalai et al., 2018; Mishra et al., 2018); 2) learn a meta-optimizer which can quickly optimize the model parameters (Ravi & Larochelle, 2016; Andrychowicz et al., 2016; Li & Malik, 2016); 3) learn an effective distance metric between examples (Snell et al., 2017; Vinyals et al., 2016; Yang et al., 2018); 4) learn an appropriate initialization from which the model parameters can be updated within a few gradient steps (Finn et al., 2017; 2018; Lee & Choi, 2018).

HSML falls into the fourth aforementioned category named as gradient-based meta-learning. Most of the gradient-based meta-learning algorithms (Finn et al., 2017; Li et al., 2017;

Flennerhag et al., 2018) assume a globally shared initialization across all tasks, as shown in Figure 1a. To accommodate dynamically changing tasks, as illustrated in Figure 1b, recent studies tailor the global shared initialization to each task by taking advantage of probabilistic models (Finn et al., 2018; Grant et al., 2018; Yoon et al., 2018a) and incorporating task-specific information (Lee & Choi, 2018; Vuorio et al., 2018). However, our proposed HSML outlined in Figure 1c customizes the global shared initialization to each cluster using a hierarchical clustering structure, which enjoys not only knowledge customization but also generalization (e.g., between task 1 and 3). Better yet, HSML right fit to a continual learning scenario with evolving clustering structures.

## 3. Preliminaries

**The Meta-Learning Problem** Suppose that a sequence of tasks  $\{\mathcal{T}_1, \dots, \mathcal{T}_{N_t}\}$  are sampled from an environment which is a probability distribution  $\mathcal{E}$  on tasks (Baxter, 1998). In each task  $\mathcal{T}_i \sim \mathcal{E}$ , we have a few examples  $\{\mathbf{x}_{i,j}, \mathbf{y}_{i,j}\}_{j=1}^{n_{tr}}$  to constitute the training set  $\mathcal{D}_{\mathcal{T}_i}^{tr}$  and the rest as the test set  $\mathcal{D}_{\mathcal{T}_i}^{te}$ . Given a base learner  $f$  with  $\theta$  as parameters, the optimal parameters  $\theta_{\mathcal{T}_i}$  are learned to make accurate predictions, i.e.,  $f_{\theta_{\mathcal{T}_i}}(\mathbf{x}_{i,j}) \rightarrow \mathbf{y}_{i,j}$ . The effectiveness of such a base learner on  $\mathcal{D}_{\mathcal{T}_i}^{tr}$  is evaluated by the loss function  $\mathcal{L}(f_{\theta_{\mathcal{T}_i}}, \mathcal{D}_{\mathcal{T}_i}^{tr})$ , which equals the mean square error  $\sum_{(\mathbf{x}_{i,j}, \mathbf{y}_{i,j}) \in \mathcal{D}_{\mathcal{T}_i}^{tr}} \|f_{\theta_{\mathcal{T}_i}}(\mathbf{x}_{i,j}) - \mathbf{y}_{i,j}\|_2^2$  for regression problems or the cross entropy loss  $-\sum_{(\mathbf{x}_{i,j}, \mathbf{y}_{i,j}) \in \mathcal{D}_{\mathcal{T}_i}^{tr}} \log p(\mathbf{y}_{i,j} | \mathbf{x}_{i,j}, f_{\theta_{\mathcal{T}_i}})$  for classification problems.

The goal of meta-learning is to learn from previous tasks a well-generalized meta-learner  $\mathcal{M}(\cdot)$  which can facilitate the training of the base learner in a future task with a few examples. In fulfillment of this, meta-learning involves two stages, i.e., meta-training and meta-testing. During meta-training, the parameters of the base learner for all tasks, i.e.,  $\{\theta_{\mathcal{T}_i}\}_{i=1}^{N_t}$ , and the meta-learner  $\mathcal{M}(\cdot)$  are optimized alternately. In virtue of  $\mathcal{M}$ , the parameters  $\{\theta_{\mathcal{T}_i}\}_{i=1}^{N_t}$  are learned to minimize the expected empirical loss over training sets of all  $N_t$  historical tasks, i.e.,  $\min_{\{\theta_{\mathcal{T}_i}\}_{i=1}^{N_t}} \sum_{i=1}^{N_t} \mathcal{L}(\mathcal{M}(f_{\theta_{\mathcal{T}_i}}), \mathcal{D}_{\mathcal{T}_i}^{tr})$ . In turn, a well-generalized  $\mathcal{M}$  can be obtained by minimizing the expected empirical loss over test sets, i.e.,  $\min_{\mathcal{M}} \sum_{i=1}^{N_t} \mathcal{L}(\mathcal{M}(f_{\theta_{\mathcal{T}_i}}), \mathcal{D}_{\mathcal{T}_i}^{te})$ . When it comes to the meta-testing phase, provided with a future task  $\mathcal{T}_t$ , the learning effectiveness and efficiency are improved by applying the meta-learner  $\mathcal{M}$  and solving  $\min_{\theta_{\mathcal{T}_t}} \mathcal{L}(\mathcal{M}(f_{\theta_{\mathcal{T}_t}}), \mathcal{D}_{\mathcal{T}_t}^{tr})$ .

**Gradient-based Meta-Learning** Here we give an overview of the representative algorithm, model-agnostic meta-learning (MAML) (Finn et al., 2017). MAML instantiates the meta-learner  $\mathcal{M}$  as a well-generalized initialization for the parameters of a base learner from which a few gradient descent steps can be performed to reach the optimal  $\theta_{\mathcal{T}_i}$  for the task  $\mathcal{T}_i$ , which means  $\mathcal{M}(f_{\theta_{\mathcal{T}_i}}) = f_{\theta_0 - \alpha \nabla_{\theta} \mathcal{L}(f_{\theta_0}, \mathcal{D}_{\mathcal{T}_i}^{tr})}$ .

As a result, the optimization of  $\mathcal{M}$  during meta-training is formulated as (one gradient step as exemplarily):

$$\min_{\theta_0} \sum_{i=1}^{N_t} \mathcal{L}(f_{\theta_0 - \alpha \nabla_{\theta} \mathcal{L}(f_{\theta}, \mathcal{D}_{\mathcal{T}_i}^{tr}), \mathcal{D}_{\mathcal{T}_i}^{te}). \quad (1)$$

## 4. Methodology

In this section, we detail the proposed HSML algorithm whose framework is presented in Figure 2. The HSML aims to adapt the transferable knowledge learned from previous tasks, namely the initialization for parameters of the base learner in gradient based meta-learning ( $\theta_0$  here), to the task  $\mathcal{T}_i$  in a cluster-specific manner, so that the optimal parameters  $\theta_{\mathcal{T}_i}$  can be achieved in as few gradient descent steps as possible. As shown in the part (c) of Figure 2, the possibilities to adaptation are completely dictated by the hierarchical task clustering structure in part (b), and the eventual path for adaptation follows the clustering result on the task  $\mathcal{T}_i$  (i.e.,  $\theta_{B1}$ ). By this means, the HSML balances between customization and generalization: the transferable knowledge is adapted to different clusters of tasks, while it is still shared among closely related tasks pertaining to the same cluster. To perform hierarchical clustering on tasks, we learn the representation of a task using the proposed task embedding network, i.e., the part (a). Next we will introduce the three stages, i.e., *task representation learning*, *hierarchical task clustering*, and *knowledge adaptation*, respectively.

### 4.1. Task Representation Learning

Learning the representation of a task  $\mathcal{T}_i$  with the whole training set  $\mathcal{D}_{\mathcal{T}_i}^{tr}$  as input is much more challenging than the common representation learning over examples, which bears a striking similarity to the connection between sentence embeddings and word embeddings in natural language processing. Inspired by common practices in learning sentence embeddings (Conneau et al., 2017), we tackle the challenge by aggregating representations of all examples  $\{\mathbf{x}_{i,j}, \mathbf{y}_{i,j}\}_{j=1}^{n^{tr}} \in \mathcal{D}_{\mathcal{T}_i}^{tr}$ . The desiderata of an ideal aggregator include 1) high representational capacity, and 2) permutational invariance to its inputs. In light of these, we propose two candidate aggregators, i.e., pooling autoencoder aggregator (PAA) and recurrent autoencoder aggregator (RAA). **Pooling Autoencoder Aggregator** To meet the first desideratum, foremost, we resort to an autoencoder that learns highly effective representation for each example. The reconstruction loss for training the autoencoder is as follows,

$$\mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr}) = \sum_{j=1}^{n^{tr}} \|\text{FC}_{dec}(\mathbf{g}_{i,j}) - \mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr})\|_2^2, \quad (2)$$

where  $\mathbf{g}_{i,j} = \text{FC}_{enc}(\mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr}))$  is the representation for the  $j$ -th example. In order to characterize the joint distribution instead of the marginal distribution only, we use  $\mathcal{F}(\cdot, \cdot)$  to preliminarily embed both features and predictions of an

example. The definition of  $\mathcal{F}$  varies from dataset to dataset, which we will detail in supplementary material C.  $\text{FC}_{enc}$  and  $\text{FC}_{dec}$  stand for the encoder composed of a stack of fully connected layers and the decoder consisting of two fully connected layers with ReLU activation, respectively. Consequently, the aggregation satisfying the permutational invariance follows,

$$\mathbf{g}_i = \text{Pool}_{j=1}^{n^{tr}}(\mathbf{g}_{i,j}), \quad (3)$$

where  $\mathbf{g}_i \in \mathbb{R}^d$  is the desired representation of task  $\mathcal{T}_i$ . Pool denotes a max or mean pooling operator over examples.

**Recurrent Autoencoder Aggregator** Motivated by recent success of the recurrent embedding aggregation in order-invariant problems such as graph embedding (Hamilton et al., 2017), we also consider a recurrent autoencoder aggregator which demonstrates more remarkable expressivity especially for a task with few examples. Different from the pooling autoencoder, examples are sequentially fed into the recurrent autoencoder, i.e.,

$$\mathcal{F}(\mathbf{x}_{i,1}^{tr}, \mathbf{y}_{i,1}^{tr}) \rightarrow \mathbf{g}_{i,1} \rightarrow \dots \rightarrow \mathbf{g}_{i,n^{tr}} \rightarrow \mathbf{d}_{i,n^{tr}} \rightarrow \dots \rightarrow \mathbf{d}_{i,1}, \quad (4)$$

where  $\forall j, \mathbf{g}_{i,j} = \text{RNN}_{enc}(\mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr}), \mathbf{g}_{i,j-1})$  and  $\mathbf{d}_{i,j} = \text{RNN}_{dec}(\mathbf{d}_{i,j+1})$  represent the learned representation and the reconstruction of the  $j$ -th example, respectively. Here  $\text{RNN}_{enc}$  and  $\text{RNN}_{dec}$  stand for a recurrent encoder (LSTM or GRU) and a recurrent decoder, respectively. The reconstruction loss is similar to Eqn. (2), except that  $\text{FC}_{dec}(\mathbf{g}_{i,j})$  is replaced with  $\mathbf{d}_{i,j}$ . Thereupon, the task representation is aggregated over representations of all examples, i.e.,

$$\mathbf{g}_i = \frac{1}{n^{tr}} \sum_j^{n^{tr}} (\mathbf{g}_{i,j}). \quad (5)$$

Regrettably, the sequential feeding of examples makes the final task representation to be permutation sensitive, which violates the second prerequisite of an ideal aggregator. We address the problem by applying the recurrent aggregator to random permutations of examples (Hamilton et al., 2017).

### 4.2. Hierarchical Task Clustering

Given the representation of a task, we propose a hierarchical task clustering structure to locate the cluster the task belongs to. Before proceeding to detail the structure, we first explicate why the hierarchical clustering is preferred over flat clustering: a single level of task groups is likely insufficient to model complex task relationship in real-world applications; for example, to identify the cross-talks between gene expressions of multiple species, the study (Kim & Xing, 2010) suggests multi-level clustering of such gene interaction.

The hierarchical clustering, following the tradition of clustering, proceeds by alternating between two steps, i.e., assignment step and update step, in a layer-wise manner.

**Assignment step:** Each task receives a cluster assignment score on each hierarchical level, and the assignment that it

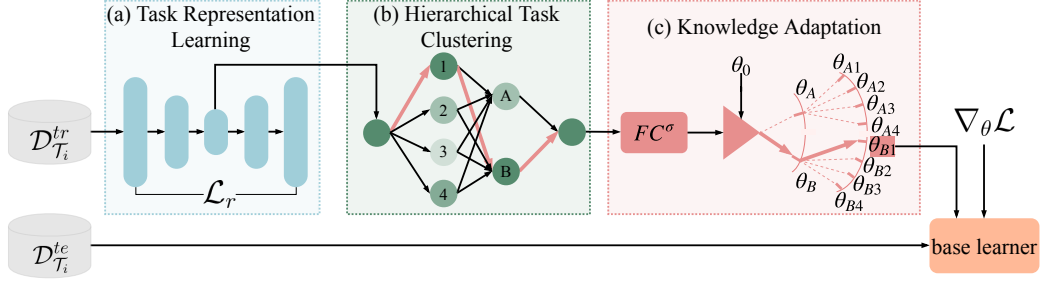


Figure 2. The framework of the proposed HSML involving three essential stages. (a) Task representation learning: we learn the representation for the task  $\mathcal{T}_i$  using an autoencoder aggregator (e.g., pooling aggregator, recurrent aggregator). (b) Hierarchical task clustering: provided with the task representation, we learn the soft clustering assignment with this differentiable hierarchical clustering structure. Darker nodes signify more likely assigned clusters (e.g., the cluster 1 in the first level and the cluster B in the second level). (c) Knowledge adaptation: we next use a parameter gate to adapt the transferable knowledge ( $\theta_0$ ) to a cluster-specific initialization ( $\theta_{B1}$ ) from which only a few gradient descent steps are required to achieve the optimal parameters  $\theta_{\mathcal{T}_i}$ .

receives in a particular level is a function of its representation in the previous level. Thus, we assign a task represented in the  $k^l$ -th cluster of the  $l$ -th level, i.e.,  $\mathbf{h}_i^{k^l} \in \mathbb{R}^d$ , to the  $k^{l+1}$ -th cluster in the  $(l+1)$ -th level. Note that we conduct soft assignment for the following two reasons: (1) task groups have been demonstrated to overlap, since there is always a continuum in the sharing between tasks (Kumar & Daumé III, 2012); (2) the soft instead of hard assignment guarantees the differentiability, so that the full HSML framework can still be trained in an end-to-end fashion. In particular, for each task  $\mathcal{T}_i$ , the soft-assignment  $p_i^{k^l \rightarrow k^{l+1}}$  is computed by applying softmax over Euclidean distances between  $\mathbf{h}_i^{k^l}$  and the learnable cluster centers  $\{\mathbf{c}_{k^{l+1}}\}_{k^{l+1}=1}^{K^{l+1}}$ , i.e.,

$$p_i^{k^l \rightarrow k^{l+1}} = \frac{\exp(-\|\mathbf{h}_i^{k^l} - \mathbf{c}_{k^{l+1}}\|_2^2 / 2\sigma^l)}{\sum_{k^{l+1}=1}^{K^{l+1}} \exp(-\|\mathbf{h}_i^{k^l} - \mathbf{c}_{k^{l+1}}\|_2^2 / 2\sigma^l)}, \quad (6)$$

where  $\sigma^l$  is a scaling factor in the  $l$ -th level and  $K^{l+1}$  denotes the number of clusters in the  $(l+1)$ -th level.

**Update step:** As a result of assignment, the representation of a task in the  $k^{l+1}$ -th cluster of the  $(l+1)$ -th level, i.e.,  $\mathbf{h}_i^{k^{l+1}}$ , can be updated with the following weighted average,

$$\mathbf{h}_i^{k^{l+1}} = \sum_{k^l=1}^{K^l} p_i^{k^l \rightarrow k^{l+1}} \tanh(\mathbf{W}^{k^{l+1}} \mathbf{h}_i^{k^l} + \mathbf{b}^{k^{l+1}}), \quad (7)$$

where  $\mathbf{W}^{k^{l+1}} \in \mathbb{R}^{d \times d}$  and  $\mathbf{b}^{k^{l+1}} \in \mathbb{R}^d$  are learned to transform from representations of the  $l$ -th to those of the  $(l+1)$ -th level.

The full pipeline of clustering starts from  $l=0$ , where the initialization for  $\mathbf{h}_i^{k^0}$  equals the task representation  $\mathbf{g}_i$  and  $K^0=1$ , and ends at  $K^L=1$ . We would especially discuss the cluster centers. The meta-learning scenario where training tasks come sequentially poses a unique challenge which requires the hierarchical clustering structure to be accordingly online. Therefore, the cluster centers are parameterized and learned as the learning proceeds. Each center is randomly initialized.

### 4.3. Knowledge Adaptation

The final representation  $\mathbf{h}_i^L$ , which encrypts the hierarchical clustering result, is believed to be cluster specific. Previous works (Xu et al., 2015; Lee & Choi, 2018) suggest that similar tasks activate similar meta-parameters (e.g., initialization) while different tasks trigger disparate ones. Inspired by this finding, we design a cluster-specific parameter gate,

$$\mathbf{o}_i = \text{FC}_{\mathbf{W}_g}^\sigma(\mathbf{g}_i \oplus \mathbf{h}_i^L), \quad (8)$$

where the fully connected layer  $\text{FC}_{\mathbf{W}_g}^\sigma$  is parameterized by  $\mathbf{W}_g$  and activated by a sigmoid function  $\sigma$ . It is worth mentioning here that concatenating the task representation  $\mathbf{g}_i$  together with  $\mathbf{h}_i^L$  not only preserves but also reinforces the cluster-specific property of the parameter gate. Most importantly, the globally transferable knowledge, i.e., the initial parameters  $\theta_0$ , is adapted to the cluster-specific initial parameters  $\theta_{0i}$  via the parameter gate, i.e.,  $\theta_{0i} = \theta_0 \circ \mathbf{o}_i$ .

Recalling the objectives for a meta-learning algorithm in Section 3, we reach the optimization problem for HSML:

$$\min_{\Theta} \sum_{i=1}^{N_t} \mathcal{L}(f_{\theta_{0i} - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_{\mathcal{T}_i}^{tr}), \mathcal{D}_{\mathcal{T}_i}^{te}}) + \xi \mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr}), \quad (9)$$

where  $\mathcal{L}$  defined in Section 3 measures the empirical risk over  $\mathcal{D}_{\mathcal{T}_i}^{te}$  and  $\mathcal{L}_r$  measures the reconstruction error as defined in Eqn. (2).  $\xi$  is used to balance the importance of these two items.  $\Theta$  represents all learnable parameters including the global transferable initialization  $\theta_0$ , the parameters for clustering, and those for knowledge adaptation (i.e.,  $\mathbf{W}_g$ ).

**Continual Adaptation** We especially pay attention to the case where a new task does not fit any of the learned task clusters, which implies that additional clusters should be introduced to the hierarchical clustering structure. Incrementally adding model capacity (Yoon et al., 2018b; Daniely et al., 2015), has been the common practice to handle distribution drift without initially introducing excessive parameters. The key lies in the criterion when to expand the clustering structure. Since the loss values of  $\mathcal{L}(f_{\theta_{\mathcal{T}_i}}, \mathcal{D}_{\mathcal{T}_i}^{te})$  fluctuate



---

**Algorithm 1** Meta-training of HSML

---

**Require:**  $\mathcal{E}$ : distribution over tasks;  $\{K^1, \dots, K^L\}$ : # of clusters in each layer;  $\alpha, \beta$ : stepsizes;  $\mu$ : threshold

- 1: Randomly initialize  $\Theta$
- 2: **while** not done **do**
- 3:   **if**  $\bar{\mathcal{L}}_{new} > \mu \bar{\mathcal{L}}_{old}$  **then**
- 4:     Increase the number of clusters
- 5:   **end if**
- 6:   Sample a batch of tasks  $\mathcal{T}_i \sim \mathcal{E}$
- 7:   **for all**  $\mathcal{T}_i$  **do**
- 8:     Sample  $\mathcal{D}_{\mathcal{T}_i}^{tr}, \mathcal{D}_{\mathcal{T}_i}^{te}$  from  $\mathcal{T}_i$
- 9:     Compute  $\mathbf{g}_i$  in Eqn. (3) or Eqn. (5),  $\mathbf{h}_i^L$  in Eqn. (7), and reconstruction error  $\mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr})$
- 10:     Compute  $\mathbf{o}_i$  in Eqn. (8) and evaluate  $\nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_{\mathcal{T}_i}^{tr})$
- 11:     Update parameters with gradient descent (taking one step as an example):  $\theta_{\mathcal{T}_i} = \theta_{0i} - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_{\mathcal{T}_i}^{tr})$
- 12:   **end for**
- 13:   Update  $\Theta \leftarrow \Theta - \beta \nabla_{\Theta} \sum_{i=1}^{N_t} \mathcal{L}(f_{\theta_{\mathcal{T}_i}}, \mathcal{D}_{\mathcal{T}_i}^{te}) + \xi \mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr})$
- 14:   Compute  $\bar{\mathcal{L}}_{new}$  and save  $\bar{\mathcal{L}}_{old}$  for every  $Q$  rounds
- 15: **end while**

---

across different tasks during the online meta-training process, setting the loss value as threshold would obviously be futile. Instead, for every  $Q$  training tasks, we compute the average loss value  $\bar{\mathcal{L}}$ . If the new average value  $\bar{\mathcal{L}}_{new}$  is more than  $\mu$  times the previous value  $\bar{\mathcal{L}}_{old}$  (i.e.,  $\bar{\mathcal{L}}_{new} > \mu \bar{\mathcal{L}}_{old}$ ), the number of clusters will be increased, and the parameters for new clusters are randomly initialized. The whole algorithm of our proposed model is detailed in Alg. 1.

## 5. Analysis

The core of HSML is to adapt a globally shared initialization of stochastic gradient descent (SGD) to be cluster specific via the proposed hierarchical clustering structure. Hence, in this section, we theoretically analyze the advantage of such adaptation in terms of the generalization bound.

For a task  $\mathcal{T}_i \sim \mathcal{E}$ , we assume both training and testing examples are i.i.d. drawn from a distribution  $\mathcal{S}_i$ , i.e.,  $\mathcal{D}_{\mathcal{T}_i}^{tr} \sim \mathcal{S}_i$  and  $\mathcal{D}_{\mathcal{T}_i}^{te} \sim \mathcal{S}_i$ . According to Theorem 2 in (Kuzborskij & Lampert, 2017), a base learner  $f_{\theta_{\mathcal{T}_i}}$  is  $\epsilon(\mathcal{S}_i, \theta_0)$ -on-average stable if its generalization is bounded by  $\epsilon(\mathcal{S}_i, \theta_0)$ , i.e.,  $\mathbb{E}_{\mathcal{S}_i} \mathbb{E}_{f_{\theta_{\mathcal{T}_i}}} [R(f_{\theta_{\mathcal{T}_i}}(\mathcal{D}_{\mathcal{T}_i}^{tr})) - \hat{R}_{\mathcal{D}_{\mathcal{T}_i}^{tr}}(f_{\theta_{\mathcal{T}_i}}(\mathcal{D}_{\mathcal{T}_i}^{tr}))] \leq \epsilon(\mathcal{S}_i, \theta_0)$ .  $\theta_0$  is the initialization of SGD to reach  $\theta_{\mathcal{T}_i}$ , and  $R(\cdot)$  and  $\hat{R}_{\mathcal{D}_{\mathcal{T}_i}^{tr}}(\cdot)$  denote the expected and empirical risk on  $\mathcal{D}_{\mathcal{T}_i}^{tr}$ , respectively.

Transferring the globally shared initialization  $\theta_0$  (i.e., MAML) to the target task  $\mathcal{T}_t$  is equivalent to transferring a hypothesis  $f_{\theta_0}$  learned from meta-training tasks like (Kuzborskij & Orabona, 2017). For HSML, the initialization can be represented as  $\theta_{0t} = \sum_{k=1}^K \hat{\mathbf{B}}_k \theta_0$ , which we demonstrate in the supplementary material A. In the following two theorems, provided with an initialization  $\theta_{0t}$

, we derive according to (Kuzborskij & Lampert, 2017) the generalization bounds of the base learner  $f_{\theta_{\mathcal{T}_t}}$  when the loss  $\mathcal{L}$  is convex and non-convex, respectively.

**Theorem 1** Assume that  $\mathcal{L}$  is convex and  $f_{\theta_{\mathcal{T}_t}}$  optimized using SGD is  $\epsilon(\mathcal{S}_t, \theta_{0t})$ -on-average stable. Then  $\epsilon(\mathcal{S}_t, \theta_{0t})$  is bounded by,

$$\mathcal{O}\left(\sqrt{\hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_{0t})} + \sqrt{\frac{1}{n^{tr}}}\right). \quad (10)$$

**Theorem 2** Assume that  $\mathcal{L} \in [0, 1]$  is  $\eta$ -smooth and has a  $\rho$ -Lipschitz Hessian. The step size at the  $u$ -step  $\alpha_u = c/u$  satisfying  $c \leq \min\{\frac{1}{\eta}, \frac{1}{4(2\eta \ln U)^2}\}$  with total steps  $U = n^{tr}$  and  $\hat{\gamma}^{\pm} = \frac{1}{n^{tr}} \sum_{j=1}^{n^{tr}} \|\nabla^2 \mathcal{L}(\theta_{0t}, (\mathbf{x}_{t,j}, \mathbf{y}_{t,j}))\|_2 + \sqrt{\hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_{0t})} \pm \frac{1}{\sqrt{n^{tr}}}$  and then  $\epsilon(\mathcal{S}_t, \theta_{0t})$  is bounded by,

$$\mathcal{O}\left(\left(1 + \frac{1}{c\hat{\gamma}^-}\right) \hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_{0t})^{\frac{c\hat{\gamma}^+}{1+c\hat{\gamma}^+}} \frac{1}{(n^{tr})^{\frac{1}{1+c\hat{\gamma}^+}}}\right). \quad (11)$$

Though some standard base learners (e.g., 4 convolutional layers in few-shot image classification (Finn et al., 2017)) with ReLU do not meet the property of Lipschitz Hessian, following (Nguyen & Hein, 2018), a softplus function  $f(x) = \frac{1}{\kappa} \log(1 + \exp(\kappa x))$  can arbitrarily well approximate ReLU by adjusting  $\kappa$  and thus Theorem 2 holds. In both cases, MAML can be regarded as the special case of HSML, i.e.,  $\forall k, \hat{\mathbf{B}}_k = \mathbf{I}$ , where  $\mathbf{I}$  is an identity matrix. Remarkably, by proving  $\exists \{\hat{\mathbf{B}}_k\}_{k=1}^K, s.t., \hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_{0t}) \leq \hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_0)$ , we conclude that HSML achieves a tighter generalization bound than MAML and thereby is much more favored. Consider the optimization process starting from  $\theta_0$ , through the negative gradient direction,  $\hat{\theta}_0 = (\mathbf{I} - \alpha \nabla \mathcal{L}(\theta_0)(\theta_0 \mathbf{I})^{-1}) \theta_0$  and  $\hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\hat{\theta}_0) \leq \hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_0)$ . Thus, we can find a  $\sum_{k=1}^K \hat{\mathbf{B}}_k = \mathbf{I} - \alpha \nabla \mathcal{L}(\theta_0)(\theta_0 \mathbf{I})^{-1}$ . We provide more details about analysis in supplementary material A.

## 6. Experiments

In this section, we evaluate the effectiveness of HSML. The goal of our experimental evaluation is to answer the following questions: (1) Can our approach outperform other meta-learning algorithms in toy regression and few-shot image classification tasks? (2) Can our approach discover reasonable task clusters? (3) Can our approach update the clustering structure in the continual learning manner and achieve better performance?

We study these questions on toy regression and few-shot image classification problems. For gradient-based meta-learning algorithms, we select the following as baselines: (1) globally shared models including MAML (Finn et al., 2017) and Meta-SGD (Li et al., 2017); (2) task specific models including MT-Net (Lee & Choi, 2018), BMAML (Yoon et al., 2018a) and MUMOMAML (Vuorio et al., 2018).

The empirical results indicate that recurrent autoencoder aggregator (RAA) is on average better than PAA for task representation, so that RAA is used as the default aggregator. We also provide a comparison of RAA and PAA on few-shot classification problem in supplementary material G. All the baselines use the same neural network structure (base learner). For hierarchical task clustering, like (Ying et al., 2018), the number of clusters in a high layer is half of that in its consecutive lower layer. We specify the hyperparameters for meta-training in supplementary material C.

## 6.1. Toy Regression

**Dataset and Experimental Settings** In the toy regression problem, different tasks are sampled from different family of functions. In this paper, the underlying family functions are (1) *Sinusoids*:  $y(x) = A \sin(wx) + b$ ,  $A \sim U[0.1, 5.0]$ ,  $w \sim U[0.8, 1.2]$  and  $b \sim U[0, 2\pi]$ ; (2) *Line*:  $y(x) = A_l x + b_l$ ,  $A_l \sim U[-3.0, 3.0]$  and  $b_l \sim U[-3.0, 3.0]$ ; (3) *Cubic*:  $y(x) = A_c x^3 + b_c x^2 + c_c x + d_c$ ,  $A_c \sim U[-0.1, 0.1]$ ,  $b_c \sim U[-0.2, 0.2]$ ,  $c_c \sim U[-2.0, 2.0]$  and  $d_c \sim U[-3.0, 3.0]$ ; (4) *Quadratic*:  $y(x) = A_q x^2 + b_q x + c_q$ ,  $A_q \sim U[-0.2, 0.2]$ ,  $b_q \sim U[-2.0, 2.0]$  and  $c_q \sim U[-3.0, 3.0]$ .  $U[\cdot, \cdot]$  represents a uniform distribution. Each individual is randomly sampled from one of the four underlying functions. The input  $x \sim U[-5.0, 5.0]$  for both training and testing tasks. We train all models for 5-shot and 10-shot regression. Mean square error (MSE) is used as evaluation metric. In hierarchical clustering, we set the number of layers to be three with 4, 2, 1 clusters in each layer, respectively.

**Results of Regression Performance** The results of 5-shot and 10-shot regression are shown in Table 1. HSML improves the performance of global models (e.g., MAML) and task specific models (e.g., MUMOMAML), indicating the effectiveness of task clustering.

Table 1. Performance of MSE  $\pm$  95% confidence intervals on toy regression tasks, averaged over 4,000 tasks. Both 5-shot and 10-shot results are reported.

Model	5-shot	10-shot
MAML	2.205 $\pm$ 0.121	0.761 $\pm$ 0.068
Meta-SGD	2.053 $\pm$ 0.117	0.836 $\pm$ 0.065
MT-Net	2.435 $\pm$ 0.130	0.967 $\pm$ 0.056
BMAML	2.016 $\pm$ 0.109	0.698 $\pm$ 0.054
MUMOMAML	1.096 $\pm$ 0.085	0.256 $\pm$ 0.028
<b>HSML (ours)</b>	<b>0.856 <math>\pm</math> 0.073</b>	<b>0.161 <math>\pm</math> 0.021</b>

**Task Clustering Analysis in Toy Regression** In order to show the power of HSML for detecting task clusters, we randomly select six tasks (more results are shown in supplementary material I) of 5-shot regression scenario and show soft-assignment values in Figure 3(a), i.e., the value of  $\{p_i^{k^0 \rightarrow k^1} | \forall k^1\}$  in Eqn. (6). Darker color stands for higher probability. The qualitative results of each task are shown in Figure 3(b). The ground truth underlying functions and the data samples  $\mathcal{D}_{T_i}^{tr}$  are shown as red lines and green stars,

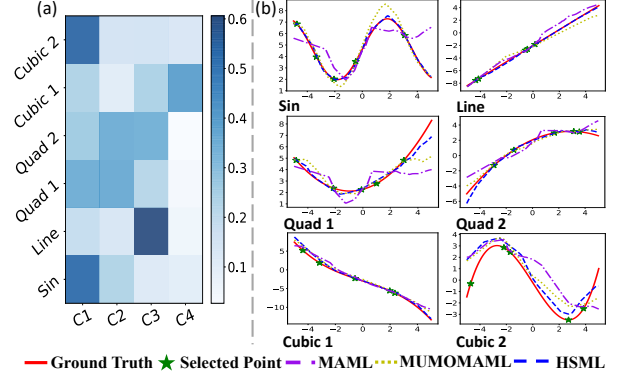


Figure 3. (a) The visualization of soft-assignment in Eqn. (6) of six selected tasks. Darker color represents higher probability. (b) The corresponding fitting curves. The ground truth underlying a function is shown in red line with data samples marked as green stars. C1-4 mean cluster 1-4, respectively.

respectively. Qualitative results of MAML, MUMOMAML (best baseline), HSML are shown in different colors.

As shown in the heatmap, sinusoids and linear with positive slope activate cluster 1 and 3, respectively. Both quadratic 1 and 2 activate cluster 2, while quadratic 1 also activates cluster 1 and quadratic 2 also activates cluster 3. From the qualitative results, we can see the shape of quadratic 2 is similar to that of linear with positive slope, while quadratic 1 has more apparent curvature. Similar findings also verify in cubic cases. The shape of cubic 2 is very similar to sinusoids, thus cluster 1 is activated. Different from cubic 2, cubic 1 mainly activates cluster 4, whose shape is similar to linear with negative slope. The results indicate that the main cluster criteria of HSML is the shapes of tasks despite the underlying family functions. Furthermore, according to the qualitative results, HSML fits better than other baselines.

**Results of Continual Adaptation** To demonstrate the effectiveness of HSML under the continual learning scenario (HSML-D), we add more underlying functions during meta-training. First, we generate tasks from sinusoids and linear, and quadratic and cubic functions are added after 15,000 and 30,000 training rounds, respectively. For comparison, one baseline is HSML with 2 fixed clusters (HSML-S(2C)), and the other is HSML with 10 fixed clusters with much more representational capability (HSML-S(10C)). The meta-training loss curve and the meta-testing performance (MSE) are shown in Figure 4. We can see that HSML-D outperforms as expected. Especially, HSML-D performs better than HSML-S(10C) which are prone to overfit and stuck at local optima at early stages.

## 6.2. Few-shot Classification

**Dataset and Experimental Settings** In the few-shot classification problem, we construct a new benchmark which currently consists of four image classification

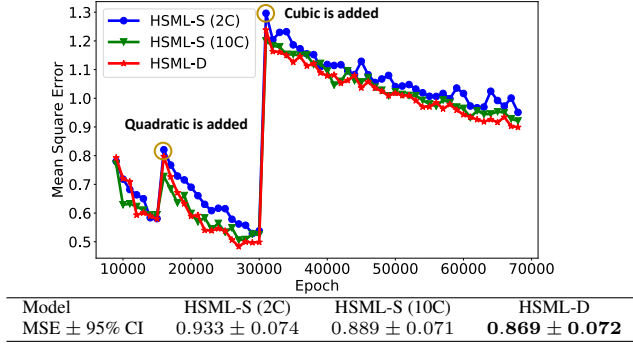


Figure 4. The performance comparison for the 5-shot toy regression problem in the continual adaptation scenario. The curve of MSE in meta-training process is shown in the top figure and the performance of meta-testing is reported in the bottom table.

datasets: Caltech-UCSD Birds-200-2011 (Bird) (Wah et al., 2011), Describable Textures Dataset (Texture) (Cimpoi et al., 2014), Fine-Grained Visual Classification of Aircraft (Aircraft) (Maji et al., 2013), and FGVCx-Fungi (Fungi) (Fun, 2018) (See supplementary material B for detailed descriptions of this benchmark). Similar to the preprocessing of MiniImagenet (Vinyals et al., 2016), we divide each dataset to meta-training, meta-validation and meta-testing classes. Following the protocol in (Finn et al., 2017; Ravi & Larochelle, 2016; Vinyals et al., 2016), we adopt N-way classification with K-shot samples. Each task samples classes from one of the four datasets. Compared with previous benchmarks (e.g., MiniImagenet) that the tasks are constructed within a single dataset, the new benchmark is more heterogeneous and closer to the real-world image classification. Like (Finn et al., 2017), the base learner is a standard four-block convolutional architecture. The number of layers in hierarchical clustering structure is set as 3 with 4, 2, 1 clusters in each layer. Note that, in this section, for the tables without confidence interval, we provide the full results in supplementary material F. In addition, we provide the comparison to MiniImagenet benchmark in supplementary material D. Note that, the sampled tasks from MiniImagenet do not have obvious heterogeneity and uncertainty. Our approach achieves comparable results among gradient-based meta-learning methods.

**Results of Classification Performance** For each dataset, we report the averaged accuracy over 1000 tasks of 5-way 1-shot/5-shot classification in Table 2. HSML consistently outperforms the other baselines on each dataset, which demonstrates the power of modeling hierarchical clustering structure. To verify the effectiveness of our proposed three components (i.e., task representation, hierarchical task clustering, knowledge adaptation), we also propose some variants of HSML. The detailed description of these variants and their corresponding results can be found in the supplementary material H, which further enhance the contribution of each component. In addition, we design another challeng-

ing leave-one-out experiment in this benchmark. We use one dataset for meta-testing and the rest three for meta-training. The results are reported in the supplementary material E and the HSML still achieves the best performance.

**Task Clustering Analysis in Few-shot Classification** Like the analysis of toy regression, we select four tasks in 5-way 1-shot classification and show their soft-assignment in Figure 5 (more results are shown in the supplementary material J). Darker color means higher probability. Furthermore, in Figure 5, we show the learned hierarchical clustering of each task. In each layer, the top activated clusters are shown in darker color and then the activation paths are generated.

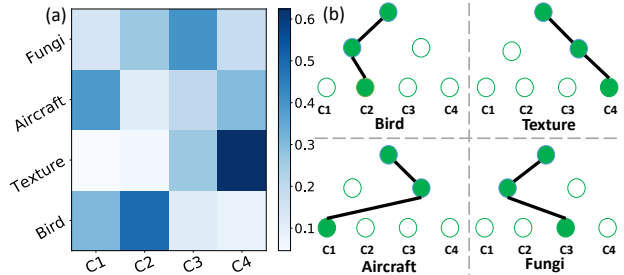


Figure 5. (a) The visualization of soft-assignment in Eqn. (6) of four selected tasks. (b) Learned hierarchical structure of each task. In each layer, top activated cluster is shown in dark color.

From Figure 5, we can see different datasets mainly activate different clusters: bird  $\rightarrow$  cluster 2, texture  $\rightarrow$  cluster 4, aircraft  $\rightarrow$  cluster 1, fungi  $\rightarrow$  cluster 3. It is also interesting to find the clustering across different tasks via the second largest activated cluster which further promote knowledge transfer between tasks. The correlation may represent the similarity of shape (bird and aircraft), environment (fungi and bird), surface texture (texture and fungi). Note that, aircraft is correlated to texture because the classification of aircraft variant is mainly based on their shape and texture. The clustering can be further verified in the learned activated path. In the second layer, the left node, which may represent the environment, is activated by cluster 2 (activated by bird) and 3 (activated by fungi). The right node that reflects surface texture is activated by cluster 1 (activated by aircraft) and 4 (activated by texture). In Figure 6, in addition, we randomly select 1000 tasks from each dataset, and show the t-SNE (Maaten & Hinton, 2008) visualization of the gated weight, i.e.,  $\theta_{0i}$ , in Eqn. (9). Compared with MUMOMAML, the results indicate that our clustering structure are able to identify the tasks in different clusters.

**Results of Continual Adaptation** In few-shot classification task, we conduct the experiments for continual adaptation in the 5-way 1-shot scenario. Initially, the tasks are generated from bird and texture datasets. Then, aircraft and fungi datasets are added after approximately meta-training round 15000 and 25000, respectively. We show the average meta-training accuracy curve and meta-testing accuracy in Figure 7, where MUMOMAML, HSML-S(2C) and HSML-

Table 2. Comparison between HSML and other gradient-based meta-learning methods on the 5-way, 1-shot/5-shot image classification problem, averaged over 1000 tasks for each dataset. Accuracy  $\pm$  95% confidence intervals are reported.

	Model	Bird	Texture	Aircraft	Fungi	Average
5-way 1-shot	MAML	53.94 $\pm$ 1.45%	31.66 $\pm$ 1.31%	51.37 $\pm$ 1.38%	42.12 $\pm$ 1.36%	44.77%
	Meta-SGD	55.58 $\pm$ 1.43%	32.38 $\pm$ 1.32%	52.99 $\pm$ 1.36%	41.74 $\pm$ 1.34%	45.67%
	MT-Net	58.72 $\pm$ 1.43%	32.80 $\pm$ 1.35%	47.72 $\pm$ 1.46%	43.11 $\pm$ 1.42%	45.59%
	BMAML	54.89 $\pm$ 1.48%	32.53 $\pm$ 1.33%	53.63 $\pm$ 1.37%	42.50 $\pm$ 1.33%	45.89%
	MUMOMAML	56.82 $\pm$ 1.49%	33.81 $\pm$ 1.36%	53.14 $\pm$ 1.39%	42.22 $\pm$ 1.40%	46.50%
	<b>HSML (ours)</b>	<b>60.98 <math>\pm</math> 1.50%</b>	<b>35.01 <math>\pm</math> 1.36%</b>	<b>57.38 <math>\pm</math> 1.40%</b>	<b>44.02 <math>\pm</math> 1.39%</b>	<b>49.35%</b>
5-way 5-shot	MAML	68.52 $\pm$ 0.79%	44.56 $\pm$ 0.68%	66.18 $\pm$ 0.71%	51.85 $\pm$ 0.85%	57.78%
	Meta-SGD	67.87 $\pm$ 0.74%	45.49 $\pm$ 0.68%	66.84 $\pm$ 0.70%	52.51 $\pm$ 0.81%	58.18%
	MT-Net	69.22 $\pm$ 0.75%	46.57 $\pm$ 0.70%	63.03 $\pm$ 0.69%	53.49 $\pm$ 0.83%	58.08%
	BMAML	69.01 $\pm$ 0.74%	46.06 $\pm$ 0.69%	65.74 $\pm$ 0.67%	52.43 $\pm$ 0.84%	58.31%
	MUMOMAML	70.49 $\pm$ 0.76%	45.89 $\pm$ 0.69%	67.31 $\pm$ 0.68%	53.96 $\pm$ 0.82%	59.41%
	<b>HSML (ours)</b>	<b>71.68 <math>\pm</math> 0.73%</b>	<b>48.08 <math>\pm</math> 0.69%</b>	<b>73.49 <math>\pm</math> 0.68%</b>	<b>56.32 <math>\pm</math> 0.80%</b>	<b>62.39%</b>

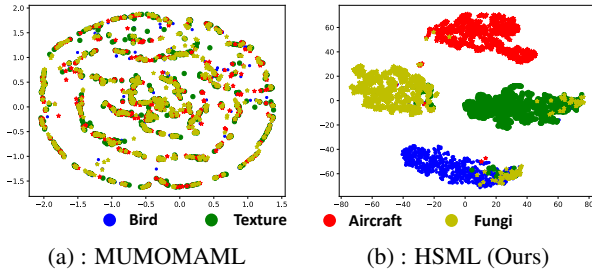
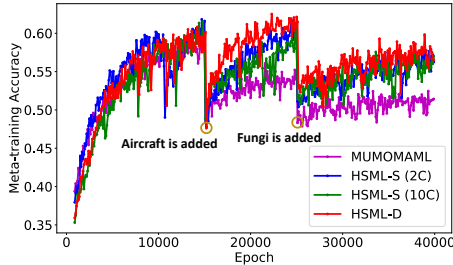


Figure 6. t-SNE visualization of gated weight, i.e.,  $\theta_{0i}$ , in Eqn. (9)

S(10C) are used as baselines. As shown in Figure 7, HSML-D consistently achieves better performance.



Model	Bird	Texture	Aircraft	Fungi
MUMOMAML	56.66%	33.68%	45.73%	40.38%
HSML-S (2C)	60.77%	33.41%	51.28%	40.78%
HSML-S (10C)	59.16%	34.48%	52.30%	40.56%
HSML-D	<b>61.16%</b>	<b>34.53%</b>	<b>54.50%</b>	<b>41.66%</b>

Figure 7. The performance comparison for the 5-way 1-shot few-shot classification problem in the continual adaptation scenario. The top figure and bottom table show the meta-training accuracy curves and the meta-testing accuracy, respectively.

**Effect of Cluster Numbers** We further analyze the effect of cluster numbers. The results are shown in Table 3. The cluster numbers from bottom layer to top layer are saved in a tuple. We can see that too few clusters may not enough to learn the task clustering characteristic (e.g., case (2,2,1)). In this dataset, increasing layers (e.g., case (8,4,4,1)) achieves

similar performance compared with case (4,2,1). However, the former introduces more parameters.

Table 3. Comparison of different cluster numbers. The numbers in first column represents the number of clusters from bottom layer to top layer. Accuracy for 5-way 1-shot classification are reported.

Num. of Clu.	Bird	Texture	Aircraft	Fungi
(2, 2, 1)	58.37%	33.18%	56.15%	42.90%
(4, 2, 1)	<b>60.98%</b>	<b>35.01%</b>	57.38%	44.02%
(6, 3, 1)	60.55%	34.02%	55.79%	43.43%
(8, 4, 2, 1)	59.55%	34.74%	<b>57.84%</b>	<b>44.18%</b>

## 7. Conclusion and Discussion

In this paper, we introduce HSML to improve the meta-learning effectiveness, which simultaneously customizing task knowledge and preserving knowledge generalization via hierarchical clustering structure. Compared with several baselines, experiments demonstrated the effectiveness and interpretability of our algorithm in both toy regression and few-shot classification problems.

Although our method is widely applicable, there are some limitations and interesting future directions. (1) In this paper, we provide a simple version for continual learning, where tasks from new underlying groups are added continually. However, to construct a more reliable lifelong learning system, it is will be necessary to consider more complex evolution relations between tasks (e.g., relationship forgetting); (2) Another interesting direction is to combining active learning with task relation learning for automatically exploring evolutionary task relations.



## Appendix

### A. Detailed Theoretical Analysis

**Proof of Theorem 1** Assuming a task  $\mathcal{T}_i$  is sampled from  $\mathcal{E}$ , its training and testing samples are i.i.d. drawn from distribution  $\mathcal{S}_i$ , i.e.,  $\mathcal{D}_{\mathcal{T}_i}^{tr} \sim \mathcal{S}_i$  and  $\mathcal{D}_{\mathcal{T}_i}^{te} \sim \mathcal{S}_i$ . According to Theorem 3 in (Kuzborskij & Lampert, 2017), if  $\mathcal{L}$  is convex, the base learner  $f_{\theta_{\mathcal{T}_i}}$  SGD is  $\epsilon(\mathcal{S}_i, \theta_0)$ -on-average-stable with

$$\epsilon(\mathcal{S}_i, \theta_0) = \mathcal{O}\left(\sqrt{c(R(\theta_0) - R^*)} \frac{\sqrt[4]{T}}{n^{tr}} + c\sigma \frac{\sqrt{T}}{n^{tr}}\right), \quad (12)$$

where  $R^* = \inf_{\theta \in \mathcal{H}} R(\theta)$ .

For a new task  $\mathcal{T}_t$ , we first prove that the initialization can be approximately represented as  $\theta_{0t} = \sum_{k=1}^K \hat{\mathbf{B}}_k \theta_0$ . Without loss of generality, here we consider a hierarchy  $C - L - 1$  in HSML.

$$\begin{aligned} \theta_{0t} &= \theta_0 \circ \mathbf{o}_t \\ &= \text{diag}(\mathbf{o}_t) \theta_0 \\ &= \text{diag}(\text{FC}_{\mathbf{W}_g}^\sigma(\mathbf{g}_t \oplus \mathbf{h}_t)) \theta_0 \\ &= \text{diag}(\text{FC}_{\mathbf{W}_g}^\sigma(\mathbf{g}_t \oplus \mathbf{h}_t)) \theta_0 \\ &\approx \text{diag}\{a_1[\mathbf{W}_g(\mathbf{g}_t \oplus \mathbf{h}_t)] + a_2\} \theta_0 \\ &= \text{diag}[\mathbf{W}'_g(\mathbf{g}_t \oplus \mathbf{h}_t) + a_2] \theta_0 \\ &= \text{diag}\left\{\mathbf{W}'_{gg}\mathbf{g}_t \oplus \mathbf{W}'_{gh} \sum_{l=1}^L p^l \tanh\left[\mathbf{W}\left(\sum_{c=1}^C p^{cl} \tanh(\mathbf{W}^l \mathbf{h}_t^c + b^l)\right) + b\right] + a_2\right\} \\ &\approx \text{diag}\left\{\mathbf{W}'_{gg}\mathbf{g}_t \oplus \mathbf{W}'_{gh} \sum_{l=1}^L p^l \left[\mathbf{W}\left(\sum_{c=1}^C p^{cl} (\mathbf{W}^l \mathbf{h}_t^c + b^l)\right) + b\right] + a_2\right\} \\ &= \text{diag}\left\{\sum_{l=1}^L \sum_{c=1}^C \left[\frac{1}{LC} \mathbf{W}'_{gg}\mathbf{g}_t \oplus p^l \mathbf{W}'_{gh} \left(p^{cl} \mathbf{W} \mathbf{W}^l \mathbf{h}_t^c + p^{cl} \mathbf{W} b^l + \frac{b}{C}\right) + \frac{a_2}{LC}\right]\right\} \theta_0 \\ &= \sum_{k=1}^K \hat{\mathbf{B}}_k \theta_0, \end{aligned} \quad (13)$$

where  $K = CL$  and  $\hat{\mathbf{B}}_{(l-1)*C+c} = \frac{1}{LC} \mathbf{W}'_{gg}\mathbf{g}_t \oplus p^l \mathbf{W}'_{gh} \left(p^{cl} \mathbf{W} \mathbf{W}^l \mathbf{h}_t^c + p^{cl} \mathbf{W} b^l + \frac{b}{C}\right) + \frac{a_2}{LC}$ . Note that the first equality holds by converting the Hadamard product into matrix multiplication, and the first and the second approximations come from first-order Taylor series of sigmoid and hybolic functions. In addition, in the  $C - L - 1$  hierarchical structure,  $\forall l, p^l = 1$ .

From Eqn. 12, we can see that  $\epsilon(\mathcal{S}_t, \theta_0)$  depends  $\sqrt{R(\theta_0)}$ . Like (Kuzborskij & Lampert, 2017), when the optimization process for task  $\mathcal{T}_t$  starts from the equivalent form that  $\theta_{0t} = \sum_{k=1}^K \hat{\mathbf{B}}_k \theta_0$ , we can bound  $\epsilon(\mathcal{S}_t, \theta_{0t})$  by using Hoeffding bound as:

$$\epsilon(\mathcal{S}_t, \theta_{0t}) \leq \mathcal{O}\left(\sqrt{\hat{R}_{\mathcal{D}_{\mathcal{T}_t}^{tr}}(\theta_{0t})} + \sqrt{\frac{1}{n^{tr}}}\right). \quad (14)$$

Thus, we reach the conclusion.

**Proof of Theorem 2** In non-convex case, we assume  $\mathcal{L}$  is  $\eta$ -smooth and has  $\rho$ -Lipschitz Hessian. According to the Corollary 1 and Proposition 1 in (Kuzborskij & Lampert, 2017), for task  $\mathcal{T}_t$ , we define:

$$\gamma = \mathcal{O}\left(\mathbb{E}_{(\mathbf{x}_{t,j}, \mathbf{y}_{t,j}) \sim \mathcal{D}_{\mathcal{T}_t}^{tr}} [\|\nabla^2 \mathcal{L}(\theta_{0t}, (\mathbf{x}_{t,j}, \mathbf{y}_{t,j}))\|_2] + \sqrt{\hat{R}(\theta_{0t})}\right), \quad (15)$$

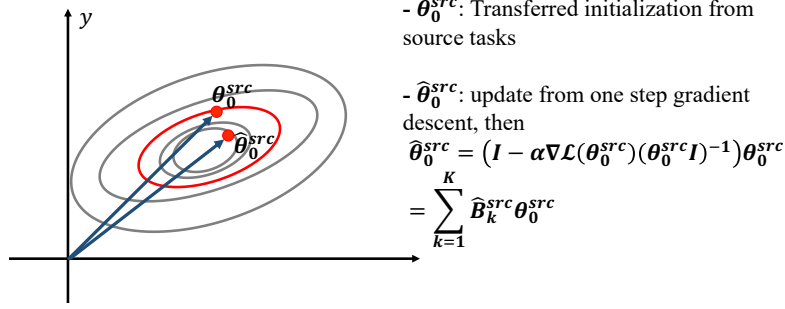


Figure 8. Illustration of Existence of  $\sum_{k=1}^K \hat{\mathbf{B}}_k$ .

and

$$\hat{\gamma} = \frac{1}{n^{tr}} \sum_{j=1}^{n^{tr}} \|\nabla^2 \mathcal{L}(\theta_{0t}, (\mathbf{x}_{t,j}, \mathbf{y}_{t,j}))\|_2 + \sqrt{\hat{R}_{\mathcal{D}_{tr}}(\theta_{0t})}. \quad (16)$$

Then, we use Hoeffding inequality and get

$$|\gamma - \hat{\gamma}| \leq \mathcal{O}\left(\frac{1}{\sqrt{n^{tr}}}\right). \quad (17)$$

Finally, let  $\hat{\gamma}^\pm = \hat{\gamma} \pm 1/\sqrt{n^{tr}}$ ,  $\epsilon(\mathcal{S}_t, \theta_{0t})$  can be bounded as:

$$\epsilon(\mathcal{S}_t, \theta_{0t}) \leq \mathcal{O}\left(\left(1 + \frac{1}{c\hat{\gamma}^-}\right) \hat{R}_{\mathcal{D}_{tr}}(\theta_{0t})^{\frac{c\hat{\gamma}^+}{1+c\hat{\gamma}^+}} \frac{1}{(n^{tr})^{\frac{1}{1+c\hat{\gamma}^+}}}\right). \quad (18)$$

Thus, we reach our conclusion.

**Existance of  $\sum_{k=1}^K \hat{\mathbf{B}}_k$**  Here, we provides more details about the analysis of existence of  $\sum_{k=1}^K \hat{\mathbf{B}}_k$ , i.e.,  $\exists \{\hat{\mathbf{B}}_k\}_{k=1}^K, s.t., \hat{R}_{\mathcal{D}_{tr}}(\theta_{0t}) \leq \hat{R}_{\mathcal{D}_{tr}}(\theta_0)$ . Though the negative gradient descent, we can get

$$\begin{aligned} \hat{\theta}_0 &= \theta_0 - \alpha \nabla \mathcal{L}_{\theta}, \\ &= (\mathbf{I} - \alpha \nabla \mathcal{L}(\theta_0)(\theta_0 \mathbf{I})^{-1}) \theta_0. \end{aligned} \quad (19)$$

Then, we can find a  $\sum_{k=1}^K \hat{\mathbf{B}}_k = \mathbf{I} - \alpha \nabla \mathcal{L}(\theta_0)(\theta_0 \mathbf{I})^{-1}$ . It can also be verified in Figure 8. Assume  $\theta_0$  is in the red contour, we can find a better parameter  $\hat{\theta}_0$  inside the contour through its negative gradient direction.

## B. Detailed Description of the New Few-shot Classification Benchmark

The new benchmark consists of four image classification datasets. All images are resized to  $84 \times 84 \times 3$ . Here, we briefly introduce each of them as follows:

- **Caltech-UCSD Birds-200-2011 (CUB-200-2011)** (Wah et al., 2011) is a bird image dataset which contains 11,788 photos of 200 bird species. In this paper, we randomly select 100 species with 60 photos in each species. We split the meta-training/meta-validation/meta-testing sets as 64/16/20 species.

- **Meta-training:** Savannah Sparrow, Dark eyed Junco, Black footed Albatross, Henslow Sparrow, Cape Glossy Starling, Black throated Sparrow, Northern Waterthrush, Hooded Warbler, Baltimore Oriole, Scarlet Tanager, Cerulean Warbler, Downy Woodpecker, Black and white Warbler, Tropical Kingbird, Canada Warbler, Blue Jay, Elegant Tern, Groove billed Ani, Mallard, European Goldfinch, Red breasted Merganser, Geococcyx, Red winged Blackbird, Ringed Kingfisher, Prairie Warbler, Florida Jay, Hooded Oriole, American Redstart, Western Wood Pewee, Sayornis, Myrtle Warbler, Yellow Warbler, Tree Swallow, Rufous Hummingbird, Fish Crow, Bewick Wren, Seaside Sparrow, Vesper Sparrow, American Crow, Eared Grebe, Blue headed Vireo, White necked Raven, Frigatebird, Horned Lark, Tree Sparrow, Red bellied Woodpecker, Pacific Loon, Caspian Tern, Anna Hummingbird, Olive sided Flycatcher, Common Tern, Cedar Waxwing, Great Crested Flycatcher, Blue Grosbeak, White breasted Kingfisher, White eyed Vireo, Purple Finch, Cliff Swallow, Scissor tailed Flycatcher, Harris Sparrow, Western Grebe, Gadwall, American Goldfinch, Pine Warbler.
- **Meta-validation:** Mockingbird, Vermilion Flycatcher, Cape May Warbler, Prothonotary Warbler, White crowned Sparrow, Ovenbird, Pomarine Jaeger, Indigo Bunting, Blue winged Warbler, Chipping Sparrow, Horned Grebe, Fox Sparrow, Green Violetear, Nashville Warbler, Least Tern, Marsh Wren.

- 
- **Meta-testing:** Rose breasted Grosbeak, Nighthawk, Long tailed Jaeger, Bronzed Cowbird, California Gull, Ivory Gull, Northern Fulmar, Brown Pelican, Ring billed Gull, Great Grey Shrike, White breasted Nuthatch, Mourning Warbler, Sage Thrasher, Horned Puffin, Pied Kingfisher, Shiny Cowbird, Scott Oriole, Red eyed Vireo, Song Sparrow, Winter Wren.
  - **Describable Textures Dataset (DTD)** (Cimpoi et al., 2014) is a texture image dataset which contains 5640 images from 47 classes. Each class contains 120 images. Meta-training/Meta-validation/Meta-testing contains 30/7/10 classes respectively.
    - **Meta-training:** pitted, woven, crosshatched, crystalline, sprinkled, lacelike, bubbly, marbled, dotted, bumpy, striped, zigzagged, lined, smeared, pleated, stratified, waffled, knitted, gauzy, porous, spiralled, grooved, banded, potholed, stained, veined, swirly, frilly, freckled, studded.
    - **Meta-validation:** wrinkled, grid, perforated, cobwebbed, honeycombed, cracked, blotchy.
    - **Meta-testing:** fibrous, matted, scaly, chequered, flecked, paisley, braided, polka-dotted, interlaced, meshed.
  - **Fine-Grained Visual Classification of Aircraft (FGVC-Aircraft)** (Maji et al., 2013) is a image dataset for fine grained visual categorization of aircraft. The dataset contains 102 different aircraft variants. In this paper, we randomly select 100 variants with 100 images in each variant. We split the meta-training/meta-validation/meta-testing to 64/16/20 variants respectively.
    - **Meta-training:** MD-90, 737-600, A310, An-12, DR-400, Falcon-900, DC-3, Challenger-600, Fokker-70, Cessna-172, 747-400, ERJ-145, Dornier-328, A330-300, A319, Model-B200, E-170, A340-500, BAE-125, Metroliner, 747-300, C-130, DH-82, Hawk-T1, 727-200, 767-300, DC-10, Spitfire, E-195, BAE-146-300, F-16A-B, Beechcraft-1900, 747-200, Boeing-717, Falcon-2000, 777-300, Cessna-560, DHC-8-100, Cessna-525, 737-200, DC-8, Global-Express, DHC-1, CRJ-200, A340-300, DC-9-30, CRJ-900, A320, 737-300, Eurofighter-Typhoon, SR-20, E-190, Saab-340, C-47, Il-76, MD-87, 757-300, DHC-6, Tu-154, 777-200, 767-200, A318, 757-200, A300B4.
    - **Meta-validation:** 737-900, A340-600, 737-800, 737-400, L-1011, A330-200, Gulfstream-V, 737-500, A340-200, ATR-72, MD-11, CRJ-700, EMB-120, Fokker-100, DC-6, 737-700.
    - **Meta-testing:** 707-320, PA-28, Cessna-208, F-A-18, DHC-8-300, ERJ-135, Tornado, BAE-146-200, A321, ATR-42, Saab-2000, Tu-134, Fokker-50, A380, MD-80, Gulfstream-IV, Yak-42, 747-100, 767-400, Embraer-Legacy-600.
  - **FGVCx-Fungi (Fungi)** (Fun, 2018) contains over 100,000 fungi images of nearly 1,500 wild mushroom species. We first filter the species with less than 150 images and then randomly select 100 species with 150 images in each species. We split the meta-training/meta-validation/meta-testing to 64/16/20 species respectively.
    - **Meta-training:** Suillus granulatus, Phaeolus schweinitzii, Cystoderma amianthinum, Pycnoporellus fulgens, Psathyrella candolleana, Meripilus giganteus, Phellinus pomaceus, Laccaria laccata, Laccaria proxima, Amanita excelsa, Ganoderma pfeifferi, Clitopilus prunulus, Agaricus arvensis, Hericium coralloides, Plicatura crispa, Agrocybe praecox, Steccherinum ochraceum, Hypholoma fasciculare, Xerocomellus pruinatus, Xerocomellus chrysenteron, Crepidotus cesatii, Auricularia auricula-judae, Heterobasidion annosum, Entoloma clypeatum, Cortinarius torvus, Mycena tintinnabulum, Laetiporus sulphureus, Datronia mollis, Pholiota squarrosa, Cerioporus squamosus, Tricholoma terreum, Coprinellus micaceus, Cyllindrobasidium laeve, Dacrymyces stillatus, Gloeophyllum sepiarium, Lycoperdon perlatum, Hygrophorus pustulatus, Clavulina coralloides, Xerocomus ferrugineus, Cortinarius albobolaceus, Byssomerulius corium, Boletus edulis, Hymenopellis radicata, Basidiaradulum radula, Cortinarius elatior, Schizophyllum commune, Cortinarius malicorius, Suillellus luridus, Ganoderma applanatum, Oligoporus guttulatus, Tubaria furfuracea, Cortinarius largus, Pleurotus ostreatus, Stereum hirsutum, Xylodon raduloides, Peniophora incarnata, Sutorius luridiformis, Flammulina velutipes var. velutipes, Phlebia radiata, Hygrocybe conica, Chlorophyllum olivieri, Armillaria ostoyae, Peniophora quercina, Mycena galericulata
    - **Meta-validation:** Agaricus impudicus, Daedaleopsis confragosa, Fomitopsis pinicola, Cortinarius anserinus, Mucidula mucida, Trametes versicolor, Stropharia cyanea, Ramaria stricta, Radulomyces confluens, Gliophorus psittacinus, Psathyrella spadiceogrisea, Coprinopsis lagopus, Daedalea quercina, Amanita muscaria, Armillaria lutea, Vuilleminia comedens
    - **Meta-testing:** Hygrocybe ceracea, Trametes hirsuta, Polyporus tuberaster, Lacrymaria lacrymabunda, Fistulina hepatica, Gymnopus dryophilus, Amanita rubescens, Fuscoporia ferrea, Craterellus undulatus, Tricholoma scalpturatum, Mycena pura, Russula depallens, Bjerkandera adusta, Trametes gibbosa, Tremella mesenterica, Cerioporus varius, Amanita fulva, Xylodon paradoxus, Cuphophyllum virgineus, Cortinarius flexipes

## C. Hyperparameters & Additional Experiment Settings

We summarize the hyperparameters in this paper in Table 4. Like (Finn et al., 2017), we compute the full Hessian-vector products for MAML. All cluster centers are randomly initialized. Note that, in few-shot classification problem, we use the change of averaged training accuracy to determine whether to increase clusters. Thus,  $\mu < 1$  in this problem. For toy regression task, the pre-aggregator embedding  $\mathcal{F}(\cdot, \cdot)$  is a fully connected layer. Following (Finn et al., 2017), the base learner has two hidden layers with 40 neurons in each. For few-shot image classification task, the pre-aggregator embedding  $\mathcal{F}(\cdot, \cdot)$  is a block of two convolutional layers with two fully connected layers. The base learner is a standard base learner

with 4 standard convolutional blocks. For continual scenario, we add one cluster every time. All the experiments are implemented using Tensorflow (Abadi et al., 2016).

Table 4. Hyperparameter summary

Hyperparameters	Toy Regreesion	miniImageNet	Multi-Datasets (New Benchmark)
Input Scale (only for image data)	/	$84 \times 84 \times 3$	$84 \times 84 \times 3$
Meta-batch Size (task batch size)	25	4	4
Inner loop learning rate ( $\alpha$ )	0.001	0.001	0.001
Outer loop learning rate ( $\beta$ )	0.001	0.01	0.01
Filters of CNN (only for image data)	/	32	32
Meta-training adaptation steps	5	5	5
Task representation size	40	128	128
Reconstruction loss weight ( $\gamma$ )	0.01	0.01	0.01
Image Embedding Size (before aggregator)	/	64	64
Continual Training Threshold ( $\tau$ )	1.25	/	0.85
# epoch (Q) for computing loss	1000	/	100

## D. Results of MiniImagenet

In this part, we present the additional comparison on MiniImagenet dataset. Similar to the analysis in (Finn et al., 2018), the sampled tasks in this benchmark do not have obvious heterogeneity and uncertainty. Thus, the goal is to compare our approach with gradient-based meta-learning methods and other previous models. The expressive capacity of each model is controlled by using 4 standard convolutional layers and the results are shown in Table 5. With the same expressive capacity, our model can achieve comparable performance with MAML-based models and other previous models in meta-learning field.

Table 5. Comparison between our approach and prior few-shot learning techniques on the 5-way, 1-shot MiniImagenet benchmark. For MT-Net (Lee & Choi, 2018), we remove the T-block since it introduces several  $1 \times 1$  convolutional layers which increases the expressive capacity of base learner (Lin et al., 2013). For BMAML (Yoon et al., 2018a), 24 classes are used for meta-testing in their original paper, while other methods use 20 classes. Since they have not released their code, we are not able to know the used classes. Thus, we implement it and report their performance on the standard classes (i.e., 20 classes for testing). Like (Finn et al., 2018), we bold methods whose highest scores that overlap in their confidence intervals.

MiniImagenet	5-way 1-shot Accuracy
Matching Nets (Vinyals et al., 2016)	$43.56 \pm 0.34\%$
meta-learner LSTM (Ravi & Larochelle, 2016)	$43.44 \pm 0.77\%$
Prototypical Network (Snell et al., 2017)	$46.61 \pm 0.78\%$
SNAIL (Mishra et al., 2018)	$45.10 \pm 0.00\%$
mAP-DLM (Triantafillou et al., 2017)	$49.82 \pm 0.78\%$
Relation Net (Yang et al., 2018)	<b><math>50.44 \pm 0.82\%</math></b>
GNN (Garcia & Bruna, 2017)	<b><math>50.33 \pm 0.36\%</math></b>
MAML (Finn et al., 2017)	$48.70 \pm 1.84\%$
LLAMA (Finn & Levine, 2017)	$49.40 \pm 1.83\%$
BMAML (Yoon et al., 2018a)	$50.01 \pm 1.86\%$
MT-Net (Lee & Choi, 2018)	$49.75 \pm 1.83\%$
MUMOMAML (Vuorio et al., 2018)	$49.86 \pm 1.85\%$
Reptile (Nichol & Schulman, 2018)	$49.97 \pm 0.32\%$
MetaSGD (Li et al., 2017)	<b><math>50.47 \pm 1.87\%</math></b>
PLATIPUS (Finn et al., 2018)	<b><math>50.13 \pm 1.86\%</math></b>
<b>HSML (ours)</b>	<b><math>50.38 \pm 1.85\%</math></b>



## E. Leave-one-out Experiments on Few-shot Image Classification

In this part, we design a more difficult experiment for few-shot image classification. For each dataset, we use three datasets for meta-training and the remaining dataset for meta-testing. For example, we use texture, bird and aircraft datasets for meta-training, and fungi dataset for meta-testing. Different from all the previous meta-learning settings which only use different classes for meta-testing, the leave-one-out experiment use a totally different dataset to test the generalization performance, which is more challenging.

The results of 5-way 1-shot classification are shown in Table 6. We compare our methods with MAML and MUMOMAML (the best baseline in few-shot classification). We can see all results are significantly worse than the results without the leave-one-out technique, which shows the difficulty of this experiment. However, by capturing task clustering structure, our method can still achieves better performance than MAML and MUMOMAML.

Table 6. Comparison of leave-one-out experiments on 5-way 1-shot classification. 4000 tasks are used to test the performance. For each dataset, the performance is reported when this dataset is used for meta-testing.

Model	Bird	Texture	Aircraft	Fungi	Average
MAML	40.76 $\pm$ 0.68%	29.50 $\pm$ 0.65%	29.54 $\pm$ 0.63%	29.94 $\pm$ 0.64%	32.43%
MUMOMAML	41.58 $\pm$ 0.68%	30.24 $\pm$ 0.68%	30.69 $\pm$ 0.66%	30.63 $\pm$ 0.66%	33.28%
HSML-RTG	<b>42.54 <math>\pm</math> 0.67%</b>	<b>30.90 <math>\pm</math> 0.67%</b>	<b>31.23 <math>\pm</math> 0.64%</b>	<b>32.98 <math>\pm</math> 0.68%</b>	<b>34.41%</b>

## F. Additional Results of Few-shot Classification

Table 7 and Table 8 contain the full results (accuracy with 95% confident interval) of few-shot image classification. Table 7 shows the full results of the bottom table in Figure 7 (in paper). Table 8 contains the full results of Table 3 (in paper).

Table 7. Comparison of online update results on few-shot image classification 5-way 1-shot scenario (Full Table).

Model	Bird	Texture	Aircraft	Fungi	Average
MUMOMAML	56.66 $\pm$ 1.43%	33.68 $\pm$ 1.37%	45.73 $\pm$ 1.39%	40.38 $\pm$ 1.40%	44.11%
HSML-Static (2C)	60.77 $\pm$ 1.43%	33.41 $\pm$ 1.40%	51.28 $\pm$ 1.37%	40.78 $\pm$ 1.34%	46.56%
HSML-Static (10C)	59.16 $\pm$ 1.49%	34.48 $\pm$ 1.36%	52.30 $\pm$ 1.35%	40.56 $\pm$ 1.39%	46.63%
HSML-Dynamic	<b>61.16 <math>\pm</math> 1.42%</b>	<b>34.53 <math>\pm</math> 1.35%</b>	<b>54.50 <math>\pm</math> 1.36%</b>	<b>41.66 <math>\pm</math> 1.41%</b>	<b>47.96%</b>

Table 8. Comparison of different cluster numbers (Full Table).

Num. of Clus.	Bird	Texture	Aircraft	Fungi	Average
(2, 2, 1)	58.37 $\pm$ 1.42%	33.18 $\pm$ 1.34%	56.15 $\pm$ 1.36%	42.90 $\pm$ 1.41%	47.65%
(4, 2, 1)	<b>60.98 <math>\pm</math> 1.50%</b>	<b>35.01 <math>\pm</math> 1.36%</b>	57.38 $\pm$ 1.40%	44.02 $\pm$ 1.39%	<b>49.35%</b>
(6, 3, 1)	60.55 $\pm$ 1.45%	34.02 $\pm$ 1.34%	55.79 $\pm$ 1.38%	43.43 $\pm$ 1.39%	48.45%
(8, 4, 4, 1)	59.55 $\pm$ 1.46%	34.74 $\pm$ 1.37%	<b>57.83 <math>\pm</math> 1.39%</b>	<b>44.18 <math>\pm</math> 1.38%</b>	49.08%

## G. Effect of Different Aggregator

In our experiment, we found that the recurrent aggregator performs the best. To give more quantitative insight about the choice of aggregator, we compare these two aggregators with different shots in Table 9. We can see that recurrent aggregator significantly outperforms in 1-shot scenario. With the increase of the size of training samples, the performances of the two aggregators become more similar. Therefore, compared with recurrent aggregator, training a better mean pooling aggregator may require more data.

## H. Ablation Studies

To investigate the contribution of different components of HSML (i.e., task representation, hierarchical task clustering, knowledge adaptation), we conduct the following ablation studies from four perspectives in Table 10, where 5-way, 1-shot results on image classification are reported. The detailed ablations are provided in follows:

- (A1) We train four MAMLs for four clusters, i.e., bird, texture, aircraft and fungi, by assigning a task to its groundtruth cluster. The results can be regarded as an upper-bound application of MAML with task clustering, provided with

Table 9. Comparison of different aggregator on different shot, where HSML-RAA and HSML-MPAA represent HSML with recurrent autoencoder aggregator and mean pooling autoencoder aggregator, respectively.

	Model	Bird	Texture	Aircraft	Fungi	Average
1-shot	HSML-MPAA	$57.87 \pm 1.48\%$	$32.07 \pm 1.36\%$	$53.76 \pm 1.41\%$	$40.88 \pm 1.37\%$	46.14%
	HSML-RAA	<b><math>60.98 \pm 1.50\%</math></b>	<b><math>35.01 \pm 1.36\%</math></b>	<b><math>57.38 \pm 1.40\%</math></b>	<b><math>44.02 \pm 1.39\%</math></b>	<b>49.35%</b>
3-shot	HSML-MPAA	$67.80 \pm 0.91\%$	$44.33 \pm 0.82\%$	$67.73 \pm 0.83\%$	$52.45 \pm 0.94\%$	58.07%
	HSML-RAA	<b><math>68.01 \pm 0.88\%</math></b>	<b><math>45.07 \pm 0.87\%</math></b>	<b><math>68.59 \pm 0.82\%</math></b>	<b><math>53.51 \pm 0.96\%</math></b>	<b>58.80%</b>
5-shot	HSML-MPAA	<b><math>71.80 \pm 0.70\%</math></b>	$48.02 \pm 0.68\%$	$71.79 \pm 0.74\%$	$54.01 \pm 0.82\%$	61.40%
	HSML-RAA	$71.68 \pm 0.73\%$	<b><math>48.08 \pm 0.69\%</math></b>	<b><math>73.49 \pm 0.68\%</math></b>	<b><math>56.32 \pm 0.80\%</math></b>	<b>62.39%</b>
8-shot	HSML-MPAA	<b><math>75.75 \pm 0.62\%</math></b>	<b><math>52.90 \pm 0.57\%</math></b>	$73.03 \pm 0.55\%$	<b><math>58.20 \pm 0.73\%</math></b>	64.97%
	HSML-RAA	$75.52 \pm 0.63\%$	$51.52 \pm 0.59\%$	<b><math>75.33 \pm 0.53\%</math></b>	$57.68 \pm 0.71\%$	<b>65.01%</b>

groundtruth clusters of all tasks which are unfortunately absent in real-world applications. HSML outperforms as the soft and hierarchical clustering not only accurately captures the task relationship but also encourages knowledge transfer across clusters.

- (A2) We investigate different variants of task representation learning in (A2a) and (A2b). In (A2a), we first use reconstruction loss to train task embeddings. Next, we fix the parameters of the task representation learning component and backpropagate meta-gradients to only train the other two components. The results are inferior, showing that meta-learning gradients further optimize task embeddings. In (A2b), we replace our task embedding with the last hidden state of the encoder. The results higher than MUMOMAML show the contribution of hierarchical clustering, while they worse than ours further justify the capability of our task embedding.
- (A3) We analyze the effect of hierarchical clustering in (A3). In (A3a), we remove the hierarchical task clustering component. In (A3b), we consider the flat instead of hierarchical task clustering. The results of (A3a) lower than (A3b) consolidate our motivation of knowledge generalization with a cluster.
- (A4) We also study three variants of knowledge adaptation in (A4a)-(A4c). In (A4a), we revise Eqn. (8) by only using the clustering representation. The results still compete with state-of-the-art baselines, but empirically the combination with the task representation yields the best performance. In (A4b), we replace the parameter gate  $\mathbf{o}_i$  with FiLM (Perez et al., 2018), where the comparable results verify the primary contribution of hierarchical clustering. In order to validate the effectiveness of parameter gate, in (A4c), we directly learn the initialization from the task representation and the cluster representation instead of using the parameter gate to mask a shared initialization. The poor results show that the parameter gate masking a shared set of parameters  $\theta_0$  may 1) prevent the curse of dimensionality and constrain the optimization space, given the high dimensionality of parameters; 2) serve as the warm-start for a new cluster of tasks in continual learning.

Table 10. Ablation Studies. Results of 5-way, 1-shot image classification are reported.

Ablation	Bird	Texture	Aircraft	Fungi
(A1): Train a MAML for each cluster, e.g., bird, by assigning a task to its groundtruth cluster.	$58.25 \pm 1.46\%$	$34.53 \pm 1.36\%$	$55.73 \pm 1.37\%$	$43.59 \pm 1.39\%$
(A2a): Use reconstruction loss to pretrain the task representation learning component, and then fix the parameters of it and backpropagate meta-gradients to only train the other two components.	$56.97 \pm 1.44\%$	$29.12 \pm 1.30\%$	$45.71 \pm 1.38\%$	$40.92 \pm 1.39\%$
(A2b): Replace our task embedding with the last hidden state of the encoder.	$58.25 \pm 1.49\%$	$34.53 \pm 1.36\%$	$55.73 \pm 1.37\%$	$43.59 \pm 1.39\%$
(A3a): Remove the hierarchical task clustering component.	$58.22 \pm 1.48\%$	$33.30 \pm 1.36\%$	$55.35 \pm 1.38\%$	$42.68 \pm 1.40\%$
(A3b): Consider only flat rather than hierarchical task clustering.	$58.08 \pm 1.45\%$	$34.26 \pm 1.35\%$	$56.11 \pm 1.38\%$	$43.38 \pm 1.39\%$
(A4a): Infer the parameter gate with the clustering representation only.	$59.01 \pm 1.50\%$	$33.69 \pm 1.35\%$	$56.69 \pm 1.39\%$	$42.88 \pm 1.40\%$
(A4b): Replace the parameter gate with FiLM (Perez et al., 2018).	$61.02 \pm 1.47\%$	$34.87 \pm 1.37\%$	$56.53 \pm 1.40\%$	$44.56 \pm 1.38\%$
(A4c): Learn the initialization directly from task and cluster representations rather than using the parameter gate.	$53.95 \pm 1.47\%$	$32.35 \pm 1.35\%$	$52.15 \pm 1.37\%$	$42.31 \pm 1.40\%$

## I. Additional Task Clustering Results of Toy Regression Tasks

In Figure I, we show the additional results of task clustering analysis of toy regression. In this figure, we further verify that tasks can be clustered by their shapes. Clusters 1 reflects the fluctuation mode curve (e.g., Sin a1-a4, Cubic a1-a4), while cluster 2 reflects an arc (e.g., Quad a2-a4). Cluster 3 mainly reflects a linear shape with positive slope (e.g. Line a1, Line a2, Quad a1, Quad a2, Cubic a1). Cluster 4 mainly reflects a linear shape with negative slope (e.g., Line a3, Line a4, Cubic a4).

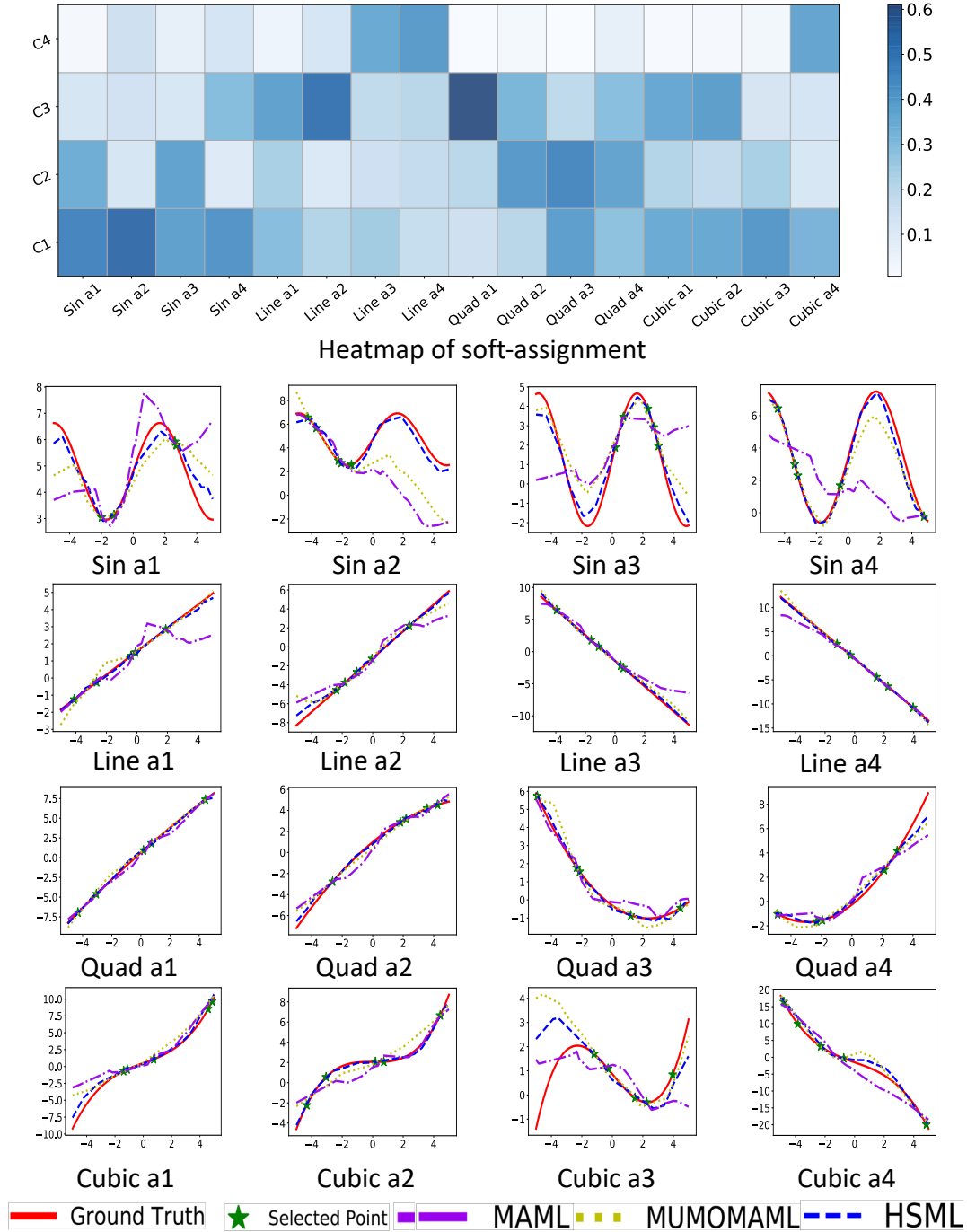


Figure 9. Additional results of task clustering analysis of toy regression problem.



## J. Additional Task Clustering Analysis of Few-shot Classification

In Figure 10, we show the additional results of task clustering analysis. The soft-assignment heatmap with their training images and activation paths of twelve tasks are illustrated. The conclusion is similar to that we draw previously in the paper. Tasks from different datasets mainly activate different clusters: bird→cluster 2, texture→cluster 4, aircraft→cluster 1, fungi→cluster 3. The left cluster and right cluster in the second layer may represent environment and surface texture, respectively.

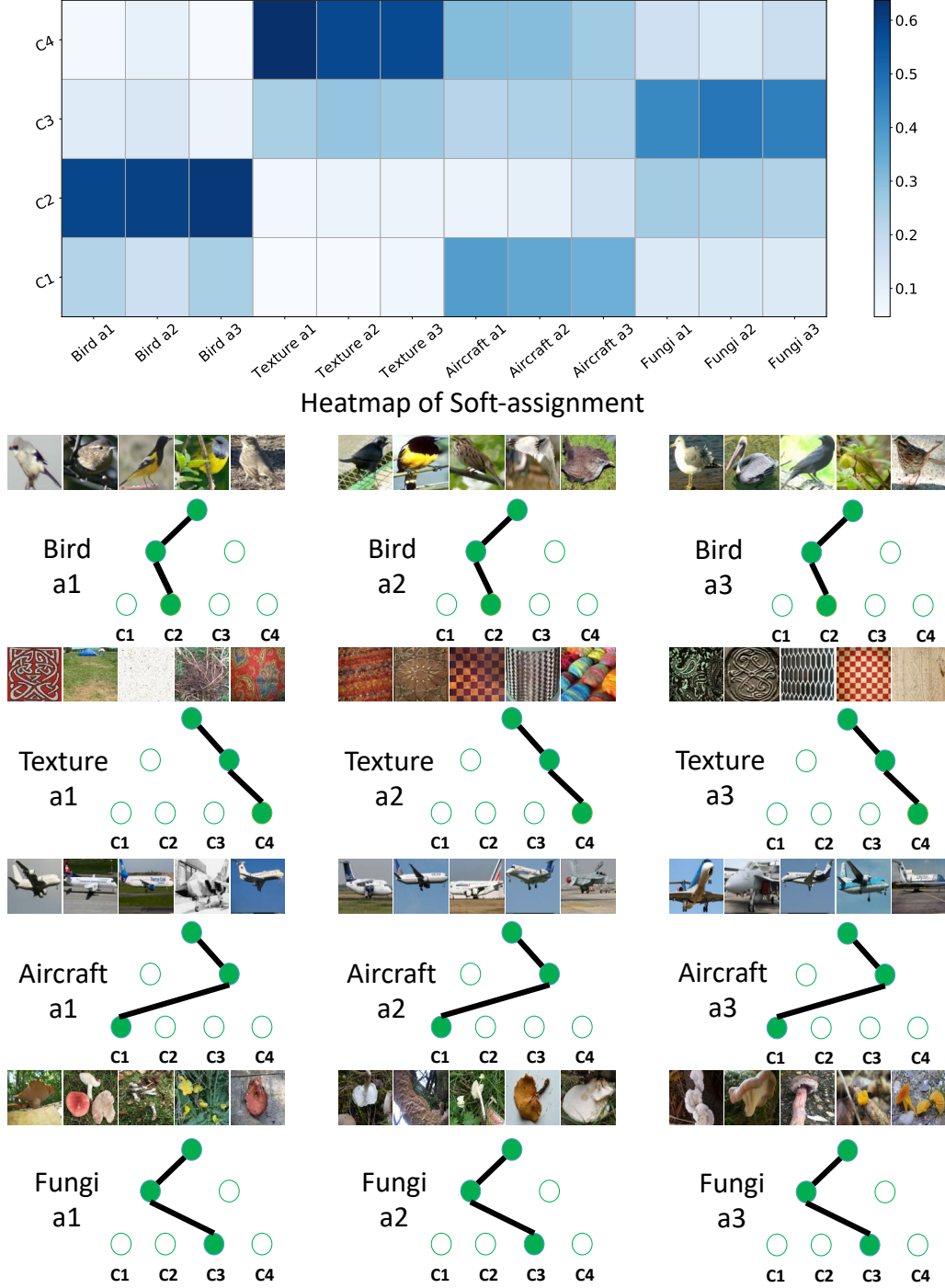


Figure 10. Additional results of task clustering analysis of few-shot image classification problem.

---

## References

- 2018 fgcvx fungi classification challenge, 2018. URL <https://www.kaggle.com/c/fungi-challenge-fgvc-2018>.
- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.
- Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N. Learning to learn by gradient descent by gradient descent. In *NIPS*, pp. 3981–3989, 2016.
- Baxter, J. Theoretical models of learning to learn. In *Learning to learn*, pp. 71–94. Springer, 1998.
- Braun, D. A., Mehring, C., and Wolpert, D. M. Structure learning in action. *Behavioural brain research*, 206(2): 157–165, 2010.
- Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., , and Vedaldi, A. Describing textures in the wild. In *CVPR*, 2014.
- Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, pp. 670–680, 2017.
- Daniely, A., Gonen, A., and Shalev-Shwartz, S. Strongly adaptive online learning. In *ICML*, pp. 1405–1411, 2015.
- Finn, C. and Levine, S. Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm. *arXiv preprint arXiv:1710.11622*, 2017.
- 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. *arXiv preprint arXiv:1806.02817*, 2018.
- Flennerhag, S., Moreno, P. G., Lawrence, N. D., and Dami-anou, A. Transferring knowledge across learning processes. *arXiv preprint arXiv:1812.01054*, 2018.
- Garcia, V. and Bruna, J. Few-shot learning with graph neural networks. *arXiv preprint arXiv:1711.04043*, 2017.
- Gershman, S. J., Blei, D. M., and Niv, Y. Context, learning, and extinction. *Psychological review*, 117(1):197, 2010.
- Gershman, S. J., Radulescu, A., Norman, K. A., and Niv, Y. Statistical computations underlying the dynamics of memory updating. *PLoS computational biology*, 10(11): e1003939, 2014.
- Grant, E., Finn, C., Levine, S., Darrell, T., and Griffiths, T. Recasting gradient-based meta-learning as hierarchical bayes. *arXiv preprint arXiv:1801.08930*, 2018.
- Gu, J., Wang, Y., Chen, Y., Cho, K., and Li, V. O. Meta-learning for low-resource neural machine translation. *arXiv preprint arXiv:1808.08437*, 2018.
- Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *NIPS*, pp. 1024–1034, 2017.
- Kim, S. and Xing, E. P. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, pp. 543–550, 2010.
- Kumar, A. and Daumé III, H. Learning task grouping and overlap in multi-task learning. In *ICML*, pp. 1723–1730, 2012.
- Kuzborskij, I. and Lampert, C. H. Data-dependent stability of stochastic gradient descent. *arXiv preprint arXiv:1703.01678*, 2017.
- Kuzborskij, I. and Orabona, F. Fast rates by transferring from auxiliary hypotheses. *Machine Learning*, 106(2): 171–195, 2017.
- Lee, Y. and Choi, S. Gradient-based meta-learning with learned layerwise metric and subspace. In *ICML*, pp. 2933–2942, 2018.
- Li, K. and Malik, J. Learning to optimize. *arXiv preprint arXiv:1606.01885*, 2016.
- 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.
- Lin, M., Chen, Q., and Yan, S. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.
- Maaten, L. v. d. and Hinton, G. Visualizing data using t-sne. *JMLR*, 9(Nov):2579–2605, 2008.
- Maji, S., Kannala, J., Rahtu, E., Blaschko, M., and Vedaldi, A. Fine-grained visual classification of aircraft. Technical report, 2013.
- Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *ICLR*, 2018.
- Munkhdalai, T. and Yu, H. Meta networks. In *ICML*, pp. 2554–2563, 2017.

- 
- Munkhdalai, T., Yuan, X., Mehri, S., and Trischler, A. Rapid adaptation with conditionally shifted neurons. In *ICML*, pp. 3661–3670, 2018.
- Nguyen, Q. and Hein, M. Optimization landscape and expressivity of deep cnns. In *ICML*, 2018.
- Nichol, A. and Schulman, J. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2018.
- Perez, E., Strub, F., de Vries, H., Dumoulin, V., and Courville, A. C. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018.
- Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. *ICLR*, 2016.
- Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *ICML*, pp. 1842–1850, 2016.
- Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *NIPS*, pp. 4077–4087, 2017.
- Triantafillou, E., Zemel, R., and Urtasun, R. Few-shot learning through an information retrieval lens. In *NIPS*, pp. 2255–2265, 2017.
- 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. Toward multimodal model-agnostic meta-learning. *arXiv preprint arXiv:1812.07172*, 2018.
- Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.
- Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pp. 2048–2057, 2015.
- Yang, F. S. Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.
- Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. Hierarchical graph representation learning with differentiable pooling. In *NIPS*, pp. 4805–4815, 2018.
- Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. Bayesian model-agnostic meta-learning. In *NIPS*, pp. 7343–7353, 2018a.
- Yoon, J., Yang, E., Lee, J., and Hwang, S. J. Lifelong learning with dynamically expandable networks. In *ICLR*, 2018b.