Received 4 December 2022, accepted 10 December 2022, date of publication 12 December 2022, date of current version 5 January 2023.

Digital Object Identifier 10.1109/ACCESS.2022.3228829

RESEARCH ARTICLE

Meta-Learning Amidst Heterogeneity and Ambiguity

KYEONGRYEOL GO[1], **MINGYU KIM**[2], **AND SEYOUNG YUN**[2]
[1]Superb AI Inc., Seoul 06232, South Korea
[2]Kim Jaechul Graduate School of AI, Korea Advanced Institute of Science and Technology (KAIST), Seoul 02455, South Korea

Corresponding author: Seyoung Yun (yunseyoung@kaist.ac.kr)

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [No.2022-0-00641, XVoice: Multi-Modal Voice Meta Learning]; This work was conducted by Center for Applied Research in Artificial Intelligence (CARAI) grant funded by DAPA and ADD (UD190031RD).

ABSTRACT Meta-learning aims to learn a model that can handle multiple tasks generated from an unknown but shared distribution. However, typical meta-learning algorithms have assumed the tasks to be similar such that a single meta-learner is sufficient to aggregate the variations in all aspects. In addition, there has been less consideration of uncertainty when limited information is given as context. In this paper, we devise a novel meta-learning framework, called Meta-learning Amidst Heterogeneity and Ambiguity (MAHA), that outperforms previous works in prediction based on its ability to task identification. By extensively conducting several experiments in regression and classification, we demonstrate the validity of our model, which turns out to generalize to both task heterogeneity and ambiguity.

INDEX TERMS Disentanglement, knowledge transfer, stochastic process, variational inference.

I. INTRODUCTION

Although deep learning models have shown remarkable performance in various domains, they have consistently been criticized because of their sensitivity to the amount of data [1]. Despite all available public data, the data scarcity problem is still not negligible. In many cases, the actual data that is worth analyzing is quite limited for many different reasons, for example, concerns about data privacy [2] and noisy data with anomalies [3]. Meta-learning that aims to handle multiple tasks by efficiently organizing the obtained knowledge has emerged as a way to overcome this deficiency with its adaptive behavior using a few data points [4], [5], [6].

Context-based meta-learning, also referred to as Neural Processes (NPs), has been emphasized since it can properly predict unseen data inputs without huge computation costs [6], [7], [8]. Many papers have shown that the context-based meta-learning algorithms are effective not only for synthetic regression, image-inpainting, and few-shot classification but also for reinforcement learning [9], [10], [11].

The context-based meta-learning utilizes neural networks to develop an appropriate identifier for a novel task by applying a context representation incorporating a permutation invariant set representation. Specifically, most papers have demonstrated that the predictive performance depends on how context information is well-constructed [8], [9], [12]. Early studies mainly focused on network architecture for permutation invariant set encoding; MLP layers followed by a simple aggregated operator like average or summation operation [6], [7]. Then, various attention mechanisms have been introduced to increase the capacity of the context representation [8], [9], [13]. Particularly, those gain better performance by constructing the context representation that considers the relationship between corresponding features from all the given context data points. To also acquire the translation invariance, convolutional neural processes employed convolutional neural networks and a kernel density estimation method with equally spaced additional data points for functional context representation [11], [14], [15].

The associate editor coordinating the review of this manuscript and approving it for publication was Prakasam Periasamy.

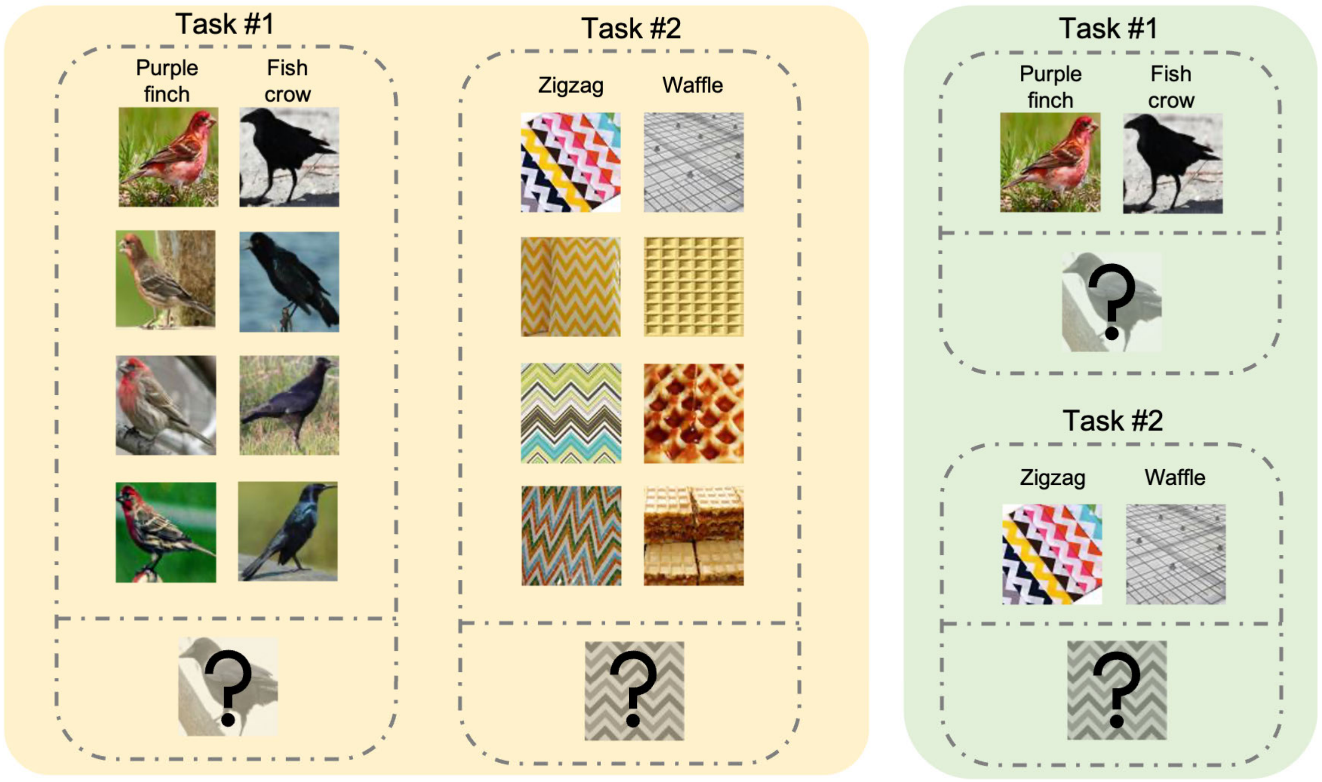


FIGURE 1. Motivating example. When confronting a task for discriminating different types of birds or textures, the data constituting the two tasks in the yellow box is very heterogeneous. Hence, there may be a limit to covering all these variations with just a single meta-learner. In the case of the green box, tasks are generated in the same way. However, only one image is provided for each class. Hence, rather than judging that the two tasks are heterogeneous, there is an incentive to misunderstand, for example, distinguishing between high and low colorfulness, which increases ambiguity on where a feature should be paid attention to.

Moreover, for noise invariance, [12], [16] proposed to apply bayesian approaches such that the bayesian aggregated function and bootstrapping method are used to improve robustness against irreducible noises.

Recently, some studies considered a novel situation in which a single context-based meta-learning serves distinct tasks simultaneously [17], [18]. These studies reported that the distinct tasks could be distinguished by explicitly designed context representation which positively influences the prediction performance of one another. Nonetheless, one significant drawback is that they supposed the number of tasks is known in advance and paid less attention to implicitly analyzing the tasks by the learned context representation itself.

In this respect, we hypothesize that a disentanglement in task representation is advantageous, frequently appearing in studies to analyze the inherent factors of variation within the dataset. This is to *i) uncover the distinctive properties as a tool for interpretability* and to *ii) explicitly separate the dataset into several clusters, which would have been detrimental when trained altogether*.

To this end, we propose a new meta-learning framework, Meta-learning Amidst Heterogeneity and Ambiguity (MAHA), that generalizes on the following two hurdles. **Task heterogeneity**: there is no clear discrimination between the tasks sampled from the faraway modes of task distribution [19], [20], [21], [22]. **Task ambiguity**: too few data points are given to infer the task identity [23], [24], [25]. (As a more direct example, see Fig. 1.) Specifically, we propose a novel method to automatically cluster between tasks and analyze their embedding by incorporating interpretable task representation with the NPs. We emphasize that the number of clusters is not predefined, which was given as a hyperparameter in previous studies to utilize complex graph structures or task-awareness relational structures.

To summarize, the main contributions of this paper are the following 3-folds:

- We construct well-clustered and interpretable context representations within task heterogeneity and ambiguity.
- We devise an optional regularization term based on the knowledge distillation technique to handle better the ambiguity for the low-*shot* regime.
- We validate MAHA through regression and classification, by which the experimental results demonstrate its ability to cope with task heterogeneity and ambiguity.

II. RELATED WORK

A. GRADIENT-BASED META-LEARNING

Meta-learning, also known as learning to learn, is to well adapt or generalize to potentially unseen tasks that are not encountered during training time. Compared to metric-based [4], [26], [27], [28] and model-based approaches [29], [30], [31], the model-agnostic nature plays a role as an incentive to use the gradient-based approaches so that a model that learns through gradient descent can be easily extended [5], [24], [32].

Many variants have emerged to balance generalization and customization in a task-adaptive manner. To begin with a generalization perspective, bayesian inspired approaches have been suggested for probabilistic extensions of meta-learning [33], [34]. In other aspects, several studies have revealed that meta-learning suffers from memorization due to a lack of training data points and suggested diverse regularization techniques [35]. From a customization perspective, either the number of parameters to adapt is reduced [36] or the auxiliary networks additionally modulate the initial parameter before the inner-loop [20], [21]. However, the deeper the network becomes, the wider the parameter space becomes, making the gradient-based meta-learning more challenging to ensure the performance of task adaptation as intended.

B. CONTEXT-BASED META-LEARNING

The family of NPs, a newly emerging framework, enables fast task adaptation as it is known to infer the contextual information of tasks well. Although it was originally devised to imitate the flexibility of the Gaussian Process (GP) [37] while resolving the scalability issue, it frequently appears as a baseline to meta-regression problem due to its adaptability [14]. Such adaptability comes from the implicit kernel learning for Bayesian inference, which is different from the typical deep kernel learning methods [38], [39], [40].

Nonetheless, many problems remain unsolved. First, the NPs rely on a complex feature extractor to enable task-specific modulation, which requires various regularization techniques with additional hyperparameters [41]. Furthermore, whereas the NPs can obtain an explicit task representation, the existing approaches have paid less attention to task representation concerning interpretability. Lastly, the performance analysis has been mainly focused on regression [8], [9], [17], [42], and some are not even directly applicable for classification [16]. We demonstrate that MAHA is free from these problems by incorporating representation learning.

C. REPRESENTATION LEARNING

Representation learning allows a deep neural network to analyze the inherent factors of variation within the dataset. Variational inference techniques are mainly utilized, where an induced regularization penalty enforces the intermediate activation to construct a semantically meaningful latent space. Here, Variational Auto-encoder (VAE) [43] is a common baseline, and follow-up studies investigated latent variables that are modeled by a multi-modal distribution, such as a mixture of the Gaussians or the Dirichlet distribution, and thus each mode indicates a distinct property [44], [45]. However, these are only applicable when the number of modes is known in advance. While [46] and [47] freed

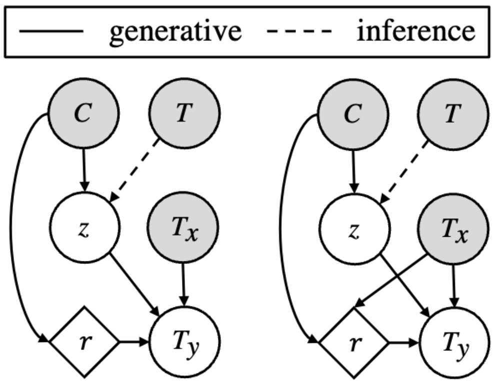


FIGURE 2. Graphical model of NP(left) and ANP(right). Circles denote random variables, whereas diamonds denote deterministic variables. Shaded variables are observed during the test phase. Every in-between edge is implemented as a neural network whose output is a parameterization of a distribution in the case of a random variable.

to know such information by modeling the latent variables with Dirichlet Process, the training gets difficult, leading to collapsing behavior. Recently, [48] and [49] also consider integrating meta-learning and representation learning, but contextual representations are either based on additional information or non-parametric. In contrast, MAHA provides qualitatively and quantitatively interpretable parametric representation without any external knowledge.

III. PROBLEM SETTING

Let $C = \{C_x, C_y\}$ be the context input-output pair (in short, context set), and let $T = \{T_x, T_y\}$ be the target input-output pair (in short, target set). Both C and T are sampled from the same task $\mathcal{T}$ from an unknown task distribution $p(\mathcal{T})$, and C is a subset of T. A common goal in meta-learning is to devise an algorithm for the model $f(\cdot, \theta)$ that appropriately uses the context set C and the model parameter θ to obtain the task-specific parameter ϕ such that when T_x is given, T_y can be accurately estimated by $f(T_x; \phi)$ with high confidence. For example, in MAML [5], a task-specific parameter can be computed by using a gradient step $\phi = \theta - \alpha \cdot \nabla_\theta L(f(C_x; \theta), C_y)$. On the other hand, in CNP [6], θ and ϕ no longer share the same parameter space. Here, the model parameter is divided into an encoder and a decoder part $\theta = \{\theta_{\text{enc}}, \theta_{\text{dec}}\}$, and the task-specific parameter can be computed by the encoder output $\phi = f_{\text{enc}}(C; \theta_{\text{enc}})$. From now on, we omit θ for brevity.

For model training, θ is iteratively updated to more readily adapt to ϕ using *batch*s where each *batch* consists of multiple tasks (possibly from different clusters) that are characterized by *way* and *shot*. If we are to estimate the corresponding output of an input given K reference input-output pairs, it is called a (1-*way*) K-*shot* regression problem. On the other hand, if we are to estimate the class label of an image among N classes given K example images of each class, it is called an N-*way* K-*shot* classification problem. Note that the class labels are shuffled whenever a task instance is created, encouraging a meta-learning algorithm to learn how to classify images even when the configuration of unseen classes occurs.

IV. BACKGROUND

A. (ATTENTIVE) NEURAL PROCESS

The NPs are devised to imitate the flexibility of GP [37] while resolving the scalability issue during the inference. In Fig. 2, we summarize how a basic family of NPs has evolved in terms of the graphical model. The encoder comprises a deterministic path and stochastic path computing the task-specific parameter $\phi = \{r, z\}$. The ϕ is the parameter of the variational distributions, which we denote by [1]

$$q(r|\{X, Y\}) = \mathcal{N}(r, 0)$$
$$q(z|\{X, Y\}) = \mathcal{N}(\mu_z, 0.1 + 0.9 \cdot \text{sigmoid}(\omega_z))$$

where $\{X, Y\}$ indicates a set of input-output pairs like C or T. Note that a reparameterization trick is applied at the end of the stochastic path for differentiable non-centered parameterization.

For both paths, NP [7] is constructed by:

$$r = \text{MeanPool}_{shot}(\text{rFF}(\{X, Y\}))$$
$$[\mu_z, \omega_z] = \text{MeanPool}_{shot}(\text{rFF}(\{X, Y\}))$$

where MeanPool$(\cdot)$ is a mean-pooling operation along the subscripted dimension, rFF$(\cdot)$ can be any row-wise feedforward layer, such as Multi-Layer Perceptron (MLP), and $[\cdot]$ denotes the concatenation.

On the other hand, ANP [8] exploits the multi-head attention MultiHead$(\cdot, \cdot, \cdot)$, connecting T_x to r in graphical model, and self-attention Self$(\cdot)$, both of which are proposed in [50].

$$r = \text{MultiHead}(T_x, X, \text{Self}(\{X, Y\})) \tag{1}$$
$$[\mu_z, \omega_z] = \text{MeanPool}_{shot}(\text{Self}(\{X, Y\})) \tag{2}$$

As in NP, the value of z is the same for every *shot* of T_x. However, based on the attention score with each element of X, r is now computed in *shot*-dependent manner. Then, conditioned on the encoder outputs, r and z, with the target input T_x, the decoder computes the parameters of predictive distribution on the target output T_y:

$$[\mu_{T_y}, \omega_{T_y}] = \text{rFF}\left([T_x, r, z]\right) \tag{3}$$

where the predictive distribution is expressed as $p(T_y|T_x, r, z) = \mathcal{N}(\mu_{T_y}, 0.1+0.9\cdot\text{softplus}(\omega_{T_y}))$. Eventually, relying on the variational inference, one can obtain the loss function, which approximates the negative ELBO by replacing an intractable $p(z|C)$ with the variational distribution $q(z|C)$ following [7]:

$$\mathcal{L}_{(A)NP} = -\mathbb{E}_{q(r|C)q(z|T)}\left[\log p(T_y|T_x, r, z)\right] + \beta KL\left(q(z|T)\|q(z|C)\right) \tag{4}$$

As a result, based on the Kolmogorov extension and de-Finetti theorems, the NPs become a stochastic process that satisfies the exchangeability and consistency [7]. However, the NPs with latent variables are empirically shown to have difficulty capturing the variability of the stochastic process [42]. Empirically, we investigated this problem by illustrating the weight norm of the decoding layer right behind the latent variables $\bar{r}$ and $\bar{z}$ in Fig. 4. The information flow is concentrated on the deterministic path for r with the tendency to ignore the stochastic path for z, which is known as the information preference problem [51]. The sparsely-coded decoder implies the redundancy of the stochastic path due to the component collapsing behavior referred to in [46] and [45].

In order to handle the information asymmetry, several solutions were proposed in studies on the generative model, such as the KL annealing scheduler [52] and expressive posterior approximation [53], but these are generally not robust to changes in model architecture. Instead, we propose a simple method to avoid the redundancy of the stochastic path in Section V by encouraging it to acquire multi-modality within heterogeneity and ambiguity.

B. SET TRANSFORMER

Set Transformer [54] is proven to be a flexible function approximator that considers a high-order interaction between set elements. It can be decomposed into the following 4 attention modules:

$$\text{MAB}(A, B) = \text{LN}(H + \text{rFF}(H)) \in \mathbb{R}^{n\times d}$$
$$\text{SAB}(A) = \text{MAB}(A, A) \in \mathbb{R}^{n\times d}$$
$$\text{ISAB}_m(A) = \text{MAB}(A, \text{MAB}(I, A)) \in \mathbb{R}^{n\times d}$$
$$\text{PMA}_k(A) = \text{MAB}(S, \text{rFF}(A)) \in \mathbb{R}^{k\times d}$$

where $H = \text{LN}(A + \text{Multihead}(A, B, B))$ is a basic building block for every module. Here, $A, B \in \mathbb{R}^{n\times d}$ are random sets, and $I \in \mathbb{R}^{m\times d}$, $S \in \mathbb{R}^{k\times d}$ are additional learnable parameters. Note that the randomly initialized inducing points I in ISAB have a lower cardinality than A.

A multi-head attention block (MAB$(\cdot, \cdot)$) and set attention block (SAB$(\cdot)$) are the two main key components, which reinforce the multi-head-attention and self-attention with a layer normalization LN$(\cdot)$ and a skip connection. The induced set attention block (ISAB$_m(\cdot)$), with m inducing points, is further devised as a substitute for the SAB$(\cdot)$ in terms of computational efficiency and generalization. The output size is fixed to k by another complex pooling module: pooling by multi-head attention (PMA$_k(\cdot)$). Note that the Set Transformer can substitute the rFF$(\cdot)$ and MeanPool$_{shot}(\cdot)$ in the encoder of NPs with more flexibility by setting $k = 1$.

V. METHODOLOGY

The core idea behind MAHA is inspired by representation learning. In other words, by learning an interpretable representation of a task, clusters of similar tasks are pre-estimated, and a meta-learning model is trained separately for each cluster. Since the conventional meta-learning models have performed well on homogeneous tasks, this way of training with representation learning allows the models to be readily

[1] Note that r is deterministic with zero variance.

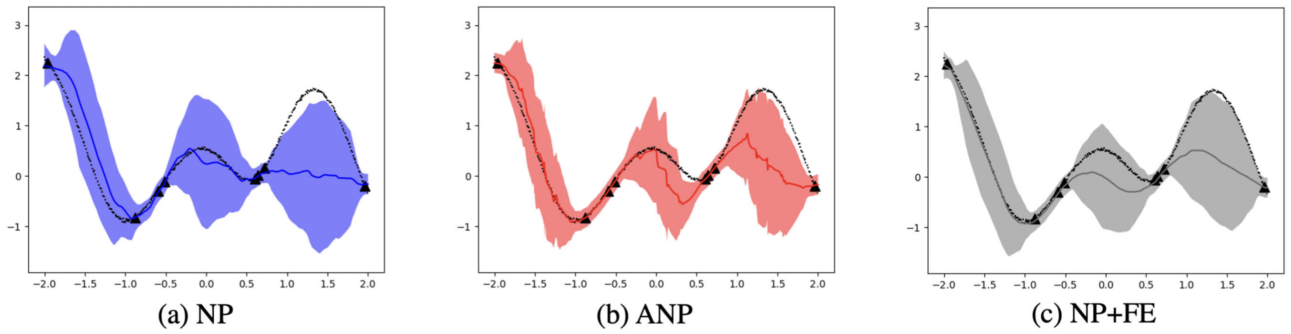


FIGURE 3. Qualitative comparison between NP, ANP, and NP with the flexible encoder (NP+FE) on functions generated from GP. The shaded areas correspond to the ±2 standard deviations. The prediction of ANP turns out to be wiggly, while NP and NP+FE are relatively smooth, following Occam's razor. Note that quantitative comparison can be looked up in Table 1.

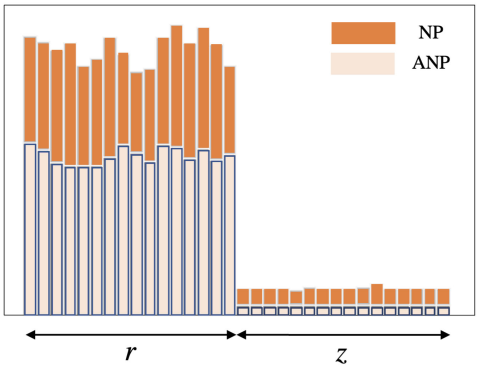


FIGURE 4. Stacked bar plot for the weight norm of the decoding layer.

extended to heterogeneous tasks. Please refer to Algorithm 1 and Section V-D for more details on the training process.

Compared to other meta-learning models, we used NPs that can explicitly obtain a microscopic representation r and a macroscopic representation z of a task. We hypothesized that task clustering would perform well if the z could contain information about task identities. To better encourage and exploit such property, we redesigned the encoder-decoder pipeline in Section V-A, and tried to improve the conventional loss function of NPs in two aspects in Section V-B and V-C. This section describes these methodologies step by step in detail.

A. ENCODER-DECODER PIPELINE

We first introduce an encoder-decoder pipeline of MAHA, namely Flexible Encoder and Linear Decoder(FELD), of which effects are examined in detail by substituting the correspondent within NP in Section VI.

1) FLEXIBLE ENCODER (FE)

Although the attention mechanism proposed in ANP was a key to resolving the underfitting in NP, there is less incentive for r to focus on task identity shared across *shot*s. As a result, in Fig. 3, ANP appears to strongly fit the given input-output pairs, which leads to a wiggly prediction. Particularly within task heterogeneity and ambiguity where the prediction space is prone to be highly variable, the wiggly prediction of ANP leads to poor interpolation and extrapolation performance. (See Fig. 6.) Therefore, in MAHA, the graphical model of NP is rather considered than that of ANP since its latent variables are *shot*-independent. Then, based on analysis in [55], the problematic underfitting is dealt with by substituting the encoder with the flexible and permutation-invariant Set Transformer [54]. In the case of computational complexity, $\mathcal{O}(mn)$ is the same as that of Set Transformer that contains a complex module, where m is the input variable number of the ISAB module, and n is $|T|$ at training and $|C|$ at testing. Note that this is comparable to that of ANP, which is $\mathcal{O}(|C||T|)$.

2) LINEAR DECODER (LD)

We avoid using a complex decoder such as [56] and weaken the complexity to allow the task-specific parameter $\phi = \{r, z\}$ to be appropriately leveraged. Specifically, we apply feature-wise linear modulation to the target input T_x. Inspired by [57], we composite the latent variables using a skip connection as the following:

$$W = \mathrm{w}(r, z) = \mathrm{LN}(r + \mathrm{rFF}(z))^{\mathrm{T}} \tag{5}$$

where the transpose operation T permutes the last two dimensions of the tensor. Among the many normalization techniques, a layer normalization [58] is applied since the statistic is computed independently for each *batch* instance such that only z can still capture the heterogeneity by the pooling proposed in Section V-B. Then, the prediction on T_y is conducted by:

$$\begin{aligned}[\mu_{T_y}, \omega_{T_y}] \text{ or } logit &= \mathrm{g}(T_x) \cdot W \\ \text{where } W &= \mathrm{w}(r, z) = \mathrm{LN}(r + \mathrm{rFF}(z))^{\mathrm{T}}\end{aligned} \tag{6}$$

where $\mathrm{g}(\cdot)$ implies any feature extractor. Note that it relates to studies on few-shot classification [41] where each column of W is computed by *shot*s within the same *way*.

B. INDUCING DISENTANGLEMENT ON z

We then introduce dimension-wise pooling and an auto-encoding structure which enable well-clustered and interpretable task representation on the stochastic path.

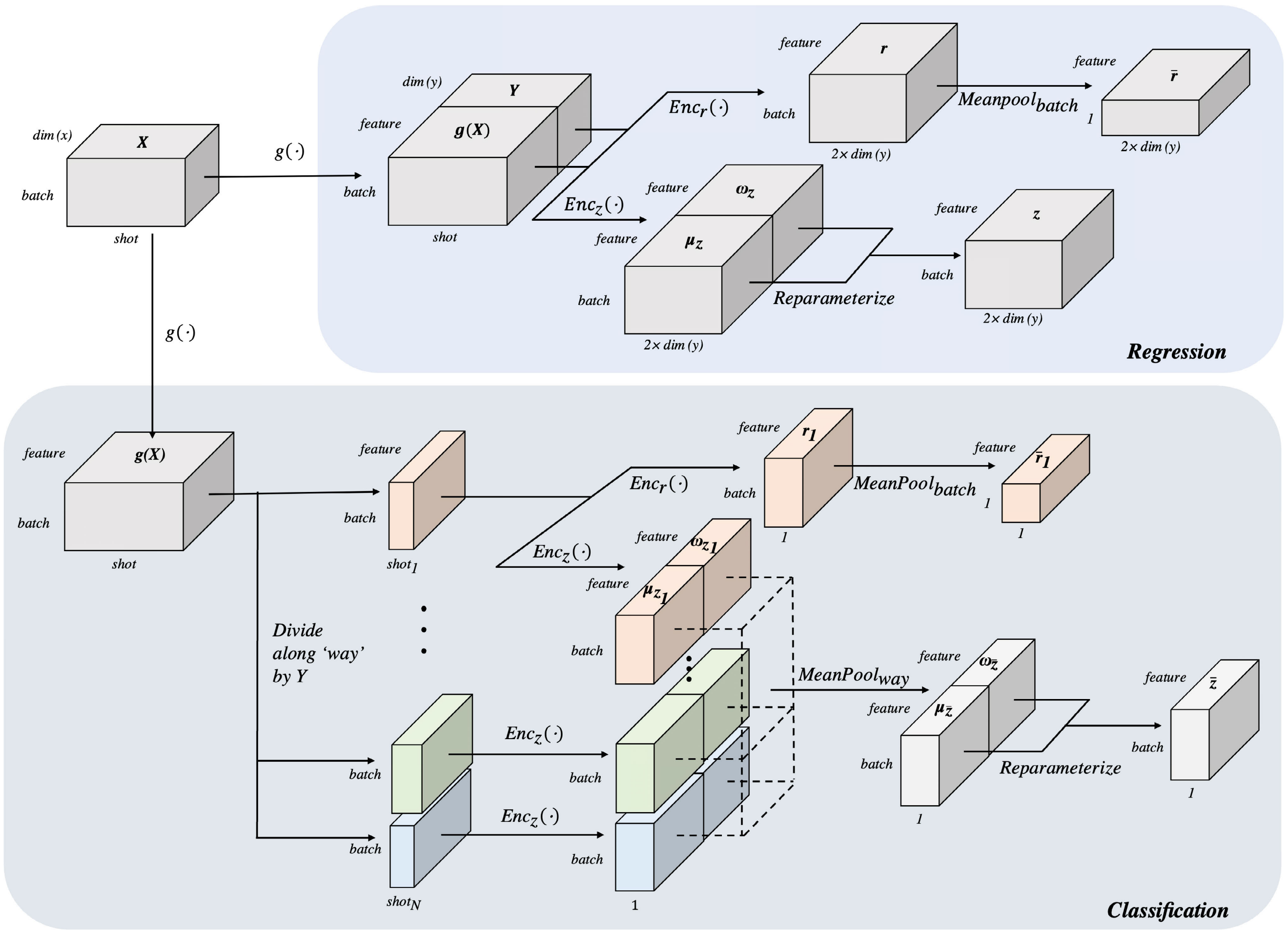


FIGURE 5. Computational diagram for $\bar{r}$ and $\bar{z}$. Note that every block of the encoder outputs in regression is reshaped from [*batch*, 1, 2 × *dim*(*y*) × *feature*] into [*batch*, 2 × *dim*(*y*), *feature*] for visual comfort. Also, the value of *way* is set to 1 in regression such that pooling on z is negligible. In classification, *shot* dimension is divided along *way* with subscript $\{1, \ldots, N\}$ and $\bar{r} = [\bar{r}_1, \ldots, \bar{r}_N]$.

1) DIMENSION-WISE POOLING

We propose a simple yet powerful architecture design to encourage the latent variables r and z to learn microscopic and macroscopic features, respectively. By pooling each path across different dimensions (*batch* for r and *way* for z), the distinct variations within the information flow are now explicitly captured:

$$\bar{r} = \text{MeanPool}_{batch}(r) \quad \text{and} \tag{7}$$

$$[\mu_{\bar{z}}, \omega_{\bar{z}}] = \text{MeanPool}_{way}([\mu_z, \omega_z]) \tag{8}$$

The $\bar{r}$ becomes identical not only across *shot*s but also across *batch*s. Therefore, whenever it is insufficient to handle all variations across tasks within the same *batch* i.e., facing task heterogeneity, the cluster-specific information can be referred only through the $\bar{z}$ since the $\bar{r}$ captures at most the average property of the clusters. On the other hand, the $\bar{z}$ allows the different *way* to share information and becomes class-invariant. Then, the columns of W from the Linear Decoder are no more independent of one another as each column is conditioned on the same $\bar{z}$ in (5), which is different from the previous studies [41]. In Fig. 5, we illustrate how the latent variables $\bar{r}$ and $\bar{z}$ are computed. Note that $\bar{r}$ is independently computed for each class, while $\bar{z}$ is pooled over the dimension of ways denoted by different colors, which provides a chance for the different classes to share information through $\bar{z}$.

2) AUTO-ENCODING STRUCTURE

Although a small subset C of T is expected to reproduce z that is obtained by T (through the KL divergence in (4)), the representation inferred by T is rather restricted to be underutilized as a side effect, which is the KL collapse [51]. By dimension-wise pooling operations, we intended to prevent z from being redundant by allowing the information flow to go through the stochastic path whenever heterogeneous tasks occur in *batch*.

Thereby, we resort to the conditional auto-encoding structure [59] on top of the dimension-wise pooling As a result, the following loss function is derived, which differs from (4) on *i) whether the pooling operations are used or not* and *ii) which set is used to compute the deterministic representation*, each of which is the result of the dimension-wise pooling and the auto-encoding structure:

$$\mathcal{L}_{pre} = -\mathbb{E}_{q(\bar{r}|T)q(\bar{z}|T)}\left[\log p(T_y|T_x, \bar{r}, \bar{z})\right] + \beta KL\left(q(\bar{z}|T)\|q(\bar{z}|C)\right) \quad (9)$$

C. DISTILLING AN OBTAINABLE KNOWLEDGE FROM T TO C

Optionally, we introduce an additional regularization term that is devised to distill an obtainable knowledge from T to C, which is to be better aware of the task ambiguity in a limited number of *shot*.

Since the output distribution accounts for a significant portion of the variability of the NPs [42], we further minimize KL divergence between the following output distributions:

$$o(T_y|C) := \mathbb{E}_{q(r|C)}\left[p(T_y|T_x, r, z)\right]$$
$$o(T_y|T) := \mathbb{E}_{q(r|T)}\left[p(T_y|T_x, r, z)\right]$$

Notice that the deterministic representations are conditioned on the different sets, one from the context set C, another from the target set T, and we assume that the stochastic representations z are given in advance. For a general purpose, we derive an upper bound as follows since the KL divergence can be computed in a closed form only in a limited family of probability distributions:

$$\begin{aligned} &KL\left(o(T_y|C)\|o(T_y|T)\right) \\ &= -\int o(T_y|C)\log o(T_y|T)\,dT_y - \mathcal{H}\left(o(T_y|C)\right) \\ &\approx -\log o(\hat{T}_y|T) - \mathcal{H}\left(o(T_y|C)\right) \quad \text{s.t.} \quad \hat{T}_y \sim o(T_y|C) \\ &\leq -\mathbb{E}_{q(r|T)}\left[\log p(\hat{T}_y|T_x, r, z)\right] - \mathcal{H}\left(o(T_y|C)\right) \end{aligned}$$

where $\mathcal{H}(\cdot)$ indicates entropy. The approximation is conducted using a Monte Carlo sample, and the inequality is from Jensen's inequality on the concave $\log(\cdot)$ function. The first term in the last line is conceptually similar to cross-entropy, which leads the model prediction to refer to the pseudo-label, which we detach from the computational graph to avoid cycle following [16]. The second term helps the model to avoid overconfidence and degeneracy, as discussed in [60]. As a result, the loss function can be rewritten as follow by augmenting the regularization term:

$$\begin{aligned} \mathcal{L}_{post} = &-\mathbb{E}_{q(r|C)q(z|T)}\left[\log p(T_y|T_x, r, z)\right] \\ &+ \beta_1 \cdot KL\left(q(z|T)\|q(z|C)\right) \\ &- \beta_2 \cdot \mathbb{E}_{q(r|T)q(z|T)}\left[\log p(\hat{T}_y|T_x, r, z)\right] \\ &- \beta_3 \cdot \mathbb{E}_{q(z|T)}\left[\mathcal{H}\left(o(T_y|C)\right)\right] \end{aligned} \quad (10)$$

Note that it can bridge to studies on knowledge distillation, specifically, a self-distillation [61], [62], where a network is trained not only with the true output T_y but also with the soft output $\hat{T}_y$ that is estimated by the network itself.

D. TRAINING PROCESS

We finally introduce the training process of MAHA, which comprises 3 steps: pre-training, clustering, and post-training.

Algorithm 1 MAHA

```
Require: Meta-train set $S^{tr}$, Meta-valid set $S^{va}$

# Step 1
Initialize the network parameter $\theta$ randomly;
while not converged in $S^{va}$ do
    for number of tasks do
        Sample a mini-batch $C$ and $T$ from $S^{tr}$;
        Update $\theta$ with $\mathcal{L}_{pre}$ in (9);
    end for
end while

# Step 2
Apply an agglomerative clustering on $q(\bar{z}|C)$;
Divide $S^{tr} = \{S^{tr}_1, \dots, S^{tr}_K\}$ based on the clusters;

# Step 3
for $k = 1, \dots, K$ do
    Initialize $\theta_k$ randomly;
    while not converged in $S^{va}$ do
        for number of tasks do
            Sample a mini-batch $C$ and $T$ from $S^{tr}_k$;
            Update $\theta_k$ with $\mathcal{L}_{(A)NP}$ in (4);
                  (or with $\mathcal{L}_{post}$ in (10))
        end for
    end while
end for
```

As for pre-training, the dimension-wise pooling and the auto-encoding structure proposed in Section V-B are used along with FELD to minimize the loss function in (9). As for clustering, agglomerative clustering is applied to the disentangled representation from the stochastic path to estimate the number of clusters with the highest purity value. For a homogeneous dataset, it can be regarded as there is only a single cluster such that the previous steps can be omitted. Please refer to Appendix B for a detailed description of the clustering process. As for post-training, for each cluster, separate FELD is trained from the beginning by (4) where the tasks are no longer uniformly sampled but statistically skewed based on the ratio of heterogeneous tasks within the cluster. Then, according to the Euclidean distance to the cluster centers, FELD, in correspondence to the closest cluster is exploited for evaluation. Note that one may optionally use (10) other than (4) to better handle the ambiguity from the low-*shot* regime. In the case of the balancing parameter, even if it is set to 1, each term operates as intended, and it is sufficient to explain the motivation of MAHA. As mentioned in beta-VAE [63], the regularization term of $\mathcal{L}_{pre}$ is important in determining the presentation quality, and the distinction term of $\mathcal{L}_{post}$ would be good to increase its influence when the gap between the context set and the target set is large.

VI. EXPERIMENT

Compared to the typical meta-learning algorithm, which has been evaluated only by homogeneous datasets, MAHA is differentiated in that it can be expanded even to heterogeneous datasets by combining representation learning. For this purpose, we intend to show the superiority of MAHA in both homogeneous and heterogeneous datasets in terms of predictive performance. We also deeply examine the roles of each methodology through empirical analysis. Compared to other studies in NPs [7], [9], [16], [17], [42], note that we considered both regression and classification referring to the problem setting in Section III. Please refer to Section VIII and Section VIII for more details on datasets and the architectural design.

Overall, we are to answer the following three questions:

- What are the benefits of using the flexible encoder and the linear decoder? (See Section VI-A.)
- How do the dimension-wise pooling and the auto-encoding structure contribute to obtaining well-clustered representation within heterogeneity and ambiguity? (See section VI-B.)
- When does the knowledge distillation become effective in terms of predictive performance? (See section VI-B.)

A. HOMOGENEOUS DATASET

To validate the proposed FELD as an encoder-decoder pipeline suitable for NP, we considered homogeneous datasets that frequently appear to evaluate meta-learning algorithms. Note that homogeneous datasets are a case in which there is only one cluster in the task representation space and can be regarded as a particular case of the heterogeneous datasets, enabling fair comparison with previous studies.

1) GAUSSIAN PROCESS

Following the basic NPs [6], [7], [8], we consider functions generated from GP with a squared exponential kernel to validate its ability to model distributions over function as a stochastic process:

$$k(x, x') = \sigma^2 \exp\left(-0.5(x - x')^2/l^2\right)$$

The experimental result in Table 1 states that although ANP performs better than NP in terms of flexibility, the dominance no longer holds when NP is equipped with the flexible encoder (NP+FE). However, a degradation in performance is shown when using the linear decoder in NP (NP+LD). This is empirical evidence that NP strongly relies on the complexity of the decoder in regression, by which the model is prone to ignore the latent variables [51]. By exploiting the flexible encoder to obtain more informative latent variables by themselves such that the (shallow) linear decoder is just enough for prediction, FELD performs better than NP and ANP, which are equipped by the (deep) conventional decoder. Moreover, it is noticeable that FELD outperforms NP+FE despite a decreased model capacity due to the linear decoder.

2) MINI-ImageNet, TIERED-ImageNet

A similar tendency can be observed in classification. We consider mini-ImageNet [26] and tiered-ImageNet [64], which

TABLE 1. MSE on gaussian process.

Model	FE	LD	MSE
NP			0.166 ± 0.002
ANP			0.142 ± 0.002
NP+FE	✓		0.138 ± 0.002
NP+LD		✓	0.312 ± 0.002
FELD	✓	✓	$\mathbf{0.130 \pm 0.002}$

are frequently used large-scale datasets for few-shot image classification. For mini-ImageNet, we follow the class split of [65], which assigns 64 classes for the meta-train set, 16 classes for the meta-valid set, and 20 classes for the meta-test set. For tiered-ImageNet, 608 classes are first grouped into 34 higher-level nodes, divided into 20, 6, and 8 nodes to construct the meta-train set, meta-valid set, and meta-test set. We use the feature provided by [24], which is obtained by pre-training a deep residual network in a supervised manner as in [28]. However, unlike [24], [66], the meta-valid set is used only for early stopping and hyperparameter search but not utilized to update the parameters.

In Table 2, 3, accuracy on mini-ImageNet and tiered-ImageNet is reported. We collect the score of various baselines that use either convolutional networks or deep residual networks and do not exploit any data augmentation for a fair comparison. While NP performs no better than a random guess when following [6], NP+LD results in a comparable score to the recent models in gradient-based meta-learning, verifying the validity of the linear decoder in classification. FELD achieves even better performance, which is remarkable in the sense that the strength of attention modules in Set Transformers can not be fully utilized in the low-*shot* regime due to the lack of referable points in the context set. This phenomenon is presumed to result from proper learning of inducing points in Set Transformer's ISAB module. Motivated from the Sparse Gaussian Process [67], the inducing points learn the pseudo-input-output pairs that are generally worth referring to for the inference. Therefore, the given context set is encoded along with such pseudo-set so the attention mechanism can be appropriately leveraged.

B. HETEROGENEOUS DATASET

We verified the validity of the key methodologies, dimension-wise pooling and auto-encoding structure, that allow the learning of interpretable task representation within heterogeneity and ambiguity. Following the experimental setting of [20], we performed the analysis in quantitative and qualitative aspects by gradually applying each of those from which a significant difference is shown in the degree of interpretability. Furthermore, we confirmed that distillation-based regularization effectively improves generalization performance under relatively high ambiguity.

1) SINE AND POLYNOMIAL

To verify the regression performance on the family of distinct functions, we experiment on the toy 1D regression as in [20]

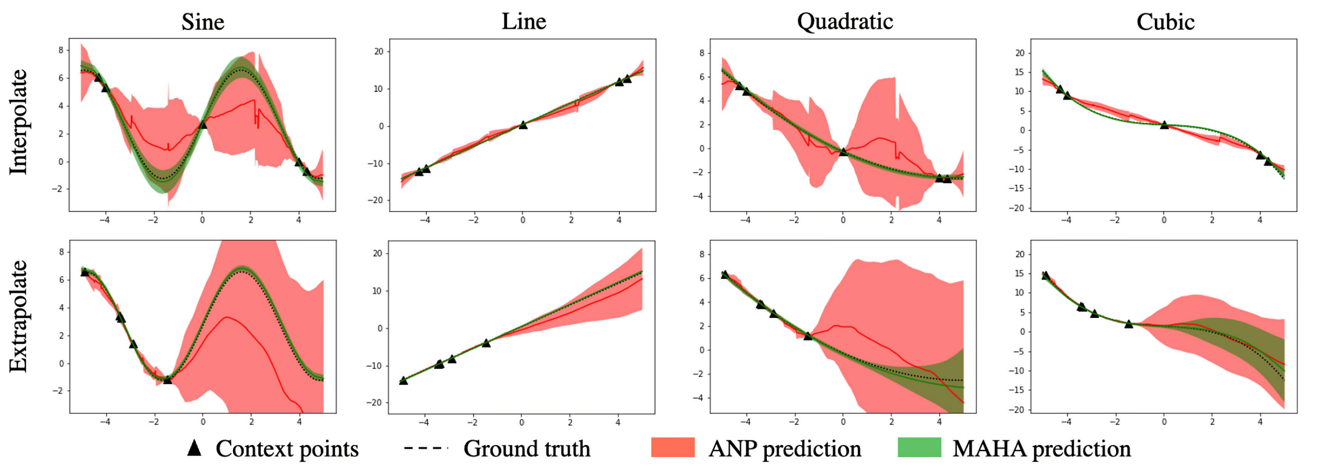


FIGURE 6. Qualitative comparison of ANP and MAHA on various function types. The context points are selected from 40% of the entire domain for extrapolation.

TABLE 2. Accuracy on mini-ImageNet.

Model	5-way 1-shot	5-way 5-shot
Matching Net	$43.40 \pm 0.78\%$	$51.09 \pm 0.71\%$
Meta-LSTM	$43.44 \pm 0.77\%$	$60.60 \pm 0.71\%$
MAML	$48.70 \pm 1.84\%$	$63.11 \pm 0.92\%$
ProtoNet	$49.42 \pm 0.78\%$	$68.20 \pm 0.66\%$
REPTILE	$49.97 \pm 0.32\%$	$65.99 \pm 0.58\%$
Relation Net	$50.44 \pm 0.82\%$	$65.32 \pm 0.70\%$
CAVIA	$51.82 \pm 0.65\%$	$65.85 \pm 0.55\%$
VERSA	$53.40 \pm 1.82\%$	$67.37 \pm 0.86\%$
TPN	$55.51 \pm 0.86\%$	$69.86 \pm 0.65\%$
Meta-SGD	$54.24 \pm 0.03\%$	$70.86 \pm 0.04\%$
SNAIL	$55.71 \pm 0.99\%$	$68.88 \pm 0.92\%$
NP+LD	$57.30 \pm 0.06\%$	$75.10 \pm 0.04\%$
TADAM	$58.50 \pm 0.30\%$	$76.70 \pm 0.30\%$
LEO	$61.76 \pm 0.08\%$	$77.59 \pm 0.12\%$
FELD	$\mathbf{62.77 \pm 0.05\%}$	$\mathbf{81.15 \pm 0.03\%}$

TABLE 3. Accuracy on tiered-ImageNet.

Model	5-way 1-shot	5-way 5-shot
MAML	$51.67 \pm 1.81\%$	$70.30 \pm 0.08\%$
ProtoNet	$53.31 \pm 0.89\%$	$72.69 \pm 0.74\%$
Relation Net	$54.48 \pm 0.93\%$	$71.32 \pm 0.78\%$
Warp-MAML	$57.20 \pm 0.90\%$	$74.10 \pm 0.70\%$
TPN	$57.41 \pm 0.94\%$	$71.55 \pm 0.74\%$
Meta-SGD	$62.95 \pm 0.03\%$	$79.34 \pm 0.06\%$
NP+LD	$63.36 \pm 0.06\%$	$80.50 \pm 0.04\%$
LEO	$66.33 \pm 0.05\%$	$81.44 \pm 0.09\%$
FELD	$\mathbf{66.87 \pm 0.06\%}$	$\mathbf{83.54 \pm 0.04\%}$

and [21]. In particular, we follow [20] where each task is randomly chosen to be one of the following one-dimensional functions where the coefficients are uniformly sampled from the prefixed intervals summarized in Section VIII:

TABLE 4. MSE on sine and polynomial.

Model	5-shot	10-shot
BMAML	2.435 ± 0.130	0.967 ± 0.056
MAML	2.205 ± 0.121	0.761 ± 0.068
META-SGD	2.053 ± 0.117	0.836 ± 0.065
MT-NET	2.016 ± 0.019	0.698 ± 0.054
MUMOMAML	1.096 ± 0.085	0.256 ± 0.028
HSML	0.856 ± 0.073	0.161 ± 0.021
NP	0.514 ± 0.051	0.089 ± 0.015
ANP	0.415 ± 0.046	0.058 ± 0.016
FELD	0.118 ± 0.015	0.008 ± 0.002
MAHA	0.077 ± 0.006	0.003 ± 0.001
MAHA*	$\mathbf{0.056 \pm 0.003}$	$\mathbf{0.002 \pm 0.001}$

$$\begin{aligned} Sine &: y = A_s sin(B_s x) + C_s \\ Line &: y = A_l x + B_l \\ Quadratic &: y = A_q x^2 + B_q x + C_q \\ Cubic &: y = A_c x^3 + B_c x^2 + C_c x + D_c \end{aligned}$$

A small number of data points are given as context, requiring the model to appropriately interpolate and extrapolate in a highly variable prediction space by correctly identifying the functional shape.

In Table 4, MSE over 4000 tasks are presented with a 95% confidence interval. Generally, all the gradient-based meta-learning algorithms are outperformed by the NPs, and a noticeable gain is again observed by exploiting the encoder-decoder pipeline, FELD. By adjusting FELD to MAHA by the pre-training and clustering process, and MAHA to MAHA* by the optional distillation term in the post-training process, a monotonic improvement is observed. In Fig. 6, we illustrate the interpolation and extrapolation of MAHA in comparison to ANP. As noted in Section V-A, the main interest of ANP is shown to fitting the context points regardless of the functional shape, which poorly performs in predicting the target outputs whose corresponding inputs are located farther away from that of the context points. This tendency can be observed during interpolation and extrapolation, leading to a wiggly prediction with significant variance. ANP's performance resulted from the failure to properly perform task identification among different task groups, resulting in overfitting to the given context points instead of proper inference on functional shape. Several studies pointed out the limitation of ANP that makes it generalizable neither to different (minor) task specifications [68] nor to different noise types [9]. By contrast, MAHA can correctly infer the functional shape, which can be confirmed by highly accurate and confident interpolation and extrapolation.

2) MULTI-DATASET

To verify the classification performance in a heterogeneous dataset, four fine-grained image classification datasets are combined to construct the multi-dataset proposed in [20]: (Bird) CUB-200-2011, (Texture) Describable Textures Dataset, (Aircraft) FGVC of Aircraft, and (Fungi) FGVCx-Fungi. In particular, an ablation study is conducted to verify the validity of the dimension-wise pooling (POOL) and the auto-encoding structure (AE) for its effect on disentanglement. Qualitatively, in Fig. 7, the mean value of the variational distribution $q(\bar{z}|C)$ is visualized through t-SNE [69]. While plausible clusters were formed except for the one that only applied POOL, it can be seen that the margin between different datasets was the largest when both POOL and AE were applied. In other words, without external knowledge, such as the number of true clusters, a combination of POOL and AE is relatively good at task identification, not only in terms of heterogeneity but also in terms of ambiguity. It is mainly due to the restricted flexibility of $\bar{r}$, which encourages $\bar{z}$ to imply not only the heterogeneity but also the local features that are initially in charge of the deterministic path. Also, the auto-encoding structure allows $\bar{r}$ to be inferred by the (large) target set T, not the (small) context set C, which is advantageous to obtain a more flexible set representation. Then, the restricted flexibility of $\bar{r}$ can be resolved so that $\bar{z}$ can provide well-clustered and interpretable task representation. The estimated purity values in Table 5 quantitatively demonstrate that the distinct datasets are not discriminated without either. Note that the validity of the methodologies stands out, particularly in the low-*shot* regime, which implies the difficulty of task identification within ambiguity.

One might wonder to what extent the purity value should reach for algorithmic success. However, since there have been limited attempts to apply representation learning to meta-learning, to the best of our knowledge, there is still no general way to measure the quality of task representation. A proper way would be devised by making linear probing [70] that frequently appears in self-supervised learning also applicable to few-shot image classification. We leave it as future work because it is beyond the scope of this paper.

TABLE 5. Purity value of the clustered task representation.

POOL	AE	1-shot	5-shot
		0.8020	0.9957
✓		0.7455	0.9145
	✓	0.9035	0.9930
✓	✓	**0.9560**	**0.9992**

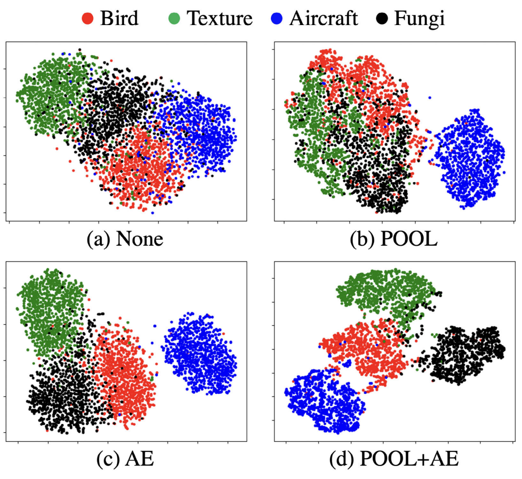


FIGURE 7. t-SNE visualization of the clustered task representation.

A similar tendency can be observed by the accuracy summarized in Table 6. Note that results on the basic NPs are omitted as it performs no better than a random guess. Noticeably, compared to the 1-*shot* setting, where a noticeable gain is occurred by adjusting FELD to MAHA, in the 5-*shot* setting, there is almost no difference between FELD and MAHA. This is because the models can identify the tasks regardless of whether the dimension-wise pooling and the auto-encoding structure are used or not, demonstrated by the sufficiently high purity values in Table 5. Accordingly, the knowledge distillation, which is fundamentally devised to regularize within ambiguity, has shown a worthwhile improvement from MAHA to MAHA*, particularly in the 1-*shot* setting. Eventually, MAHA (and MAHA*) beat all the previous works by a fairly large margin.

VII. CONCLUSION

This paper proposes a new meta-learning framework, MAHA, that generalizes amidst heterogeneity and ambiguity. We aim to disentangle the stochastic representation by the dimension-wise pooling and the auto-encoding structure based on the newly devised encoder-decoder pipeline to better leverage the latent variables. With the multi-step training process, comprehensive experiments are conducted on regression and classification. In the end, we argue that the proposed model captures the task identity with lower variance, leading to a noticeable improvement in performance. By orthogonally applying to the existing work, the compatibility and the necessity are empirically verified.

TABLE 6. Accuracy on multi-dataset.

	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%
	HSML	$60.98 \pm 1.50\%$	$35.01 \pm 1.36\%$	$57.38 \pm 1.40\%$	$44.02 \pm 1.39\%$	49.35%
	FELD	$56.17 \pm 0.64\%$	$35.86 \pm 0.41\%$	$53.03 \pm 0.58\%$	$45.41 \pm 0.58\%$	47.61%
	MAHA	$\mathbf{63.89 \pm 0.34\%}$	$37.22 \pm 0.23\%$	$\mathbf{58.90 \pm 0.44\%}$	$47.95 \pm 0.34\%$	51.99%
	MAHA*	$\mathbf{64.45 \pm 0.36\%}$	$\mathbf{37.83 \pm 0.23\%}$	$\mathbf{59.18 \pm 0.43\%}$	$\mathbf{48.33 \pm 0.33\%}$	**52.41%**
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%
	HSML	$71.68 \pm 0.73\%$	$48.08 \pm 0.69\%$	$73.49 \pm 0.68\%$	$56.32 \pm 0.80\%$	62.39%
	FELD	$\mathbf{77.63 \pm 0.46\%}$	$\mathbf{55.80 \pm 0.38\%}$	$75.88 \pm 0.41\%$	$63.68 \pm 0.50\%$	68.24%
	MAHA	$75.04 \pm 0.26\%$	$54.39 \pm 0.21\%$	$\mathbf{79.98 \pm 0.20\%}$	$\mathbf{65.09 \pm 0.25\%}$	68.62%
	MAHA*	$75.82 \pm 0.26\%$	$54.28 \pm 0.22\%$	$\mathbf{79.91 \pm 0.19\%}$	$\mathbf{65.18 \pm 0.25\%}$	**68.79%**

An interesting future work would be to apply our model to reinforcement learning. In particular, training a policy directly from well-clustered representations for sample-efficient exploration seems promising in an environment with sparse rewards.

VIII. BROADER IMPACT

When training meta-learning models, there comes a customization process based on the problem at hand. If not using the benchmark datasets frequently appearing in academia, it becomes unclear to which extent the distinct datasets should be combined, expecting the model to be versatile on every possible task generation. MAHA, in this respect, can guide for a human to analyze and cluster the available data into separate clusters. Moreover, MAHA mainly benefits future AI industries where the limited communication between the decentralized servers is available as it can infer the global context even with a small amount of information. As a result, we do not expect any negative societal impacts, but we believe that MAHA possesses many implications in more realistic scenarios.

APPENDIX A
DATASET

A. GAUSSIAN PROCESS

A batch of size 16, a context set of variable size ranged from 5 to 10, and a target set of size 30 are considered. For the squared exponential kernel

$$k(x, x') = \sigma^2 \exp\left(-0.5(x - x')^2/l^2\right)$$

the hyperparameters are chosen to be $l = 0.5$ and $\sigma = 1$. Inputs are uniformly sampled from $[-2.0, 2.0]$, and outputs are computed based on the Cholesky decomposition of the kernel with the noise parameter $\sigma_n = 0.02$ [37].

TABLE 7. Coefficient settings.

	Sine	Line	Quad	Cubic
A	[0.1, 5.0]	[-3.0, 3.0]	[-0.2, 0.2]	[-0.1, 0.1]
B	[0.8, 1.2]	[-3.0, 3.0]	[-2.0, 2.0]	[-0.2, 0.2]
C	$[0.0, 2\pi]$	-	[-3.0, 3.0]	[-2.0, 2.0]
D	-	-	-	[-3.0, 3.0]

TABLE 8. Summary of mini-ImageNet, tiered-ImageNet.

Dataset	mini-ImageNet	tiered-ImageNet
Source	[71]	[71]
Split setting	[26]	[64]
Fineness	Coarse	Coarse

B. MINI-ImageNet, TIERED-ImageNet

A batch of size 12 is considered where each batch instance is generated by sampling five random classes from the meta set with randomly assigned labels from {0, 1, 2, 3, 4}. Then, for each of the chosen classes, 1 or 5 images are selected as the context set, and 15 other images are additionally selected to construct the target set.

C. SINE AND POLYNOMIAL

A batch of size 25, a context set of size 5 or 10, and a target set of size 15 or 20 are considered. The input domain is fixed to $[-5.0, 5.0]$, and a task is defined among the four functions whose coefficients are uniformly sampled from the intervals summarized in Table 7.

D. MULTI-DATASET

Task generation process and size of the context/target set are equal to the setting in mini-ImageNet and tiered-ImageNet. However, unlike mini-ImageNet or tiered-ImageNet, images are not pre-processed in advance by the deep residual

TABLE 9. Summary of multi-dataset.

Dataset	Bird	Texture	Aircraft	Fungi
Source	[72]	[73]	[74]	[75]
Split setting	[20]	[20]	[20]	[20]
Fineness	Fine	Fine	Fine	Fine

TABLE 10. 5-*shot* multi-dataset.

Cluster index	Sine	Line	Quad	Cubic
1st	584	9	10	7
2nd	11	589	614	676

TABLE 11. 10-*shot* sine and polynomial.

Cluster index	Sine	Line	Quad	Cubic
1st	587	0	0	0
2nd	1	598	658	656

TABLE 12. 1-*shot* multi-dataset.

Cluster index	Bird	Texture	Aircraft	Fungi
1st	9	0	950	0
2nd	930	17	45	27
3rd	11	12	0	921
4th	6	1023	0	49

TABLE 13. 5-*shot* multi-dataset.

Cluster index	Bird	Texture	Aircraft	Fungi
1st	1030	0	0	0
2nd	0	0	0	979
3rd	0	1006	0	3
4th	0	0	982	0

network. Instead, all images in the meta-train set, meta-valid set, and meta-test set are resized to 84 × 84 × 3, and Conv-blocks are utilized to extract the feature from the images. Due to extensive memory usage during the feature extraction, a small batch of size 4 is considered, where each batch instance is generated among the four fine-grained image classification datasets.

APPENDIX B
AGGLOMERATIVE CLUSTERING

For the heterogeneous datasets, an agglomerative clustering is applied to the t-SNE embeddings of $\mu_{\tilde{z}}$ from the stochastic path where we use the default setting of Scipy [76], an open-source scientific tool for Python. In Table 10 to 11 and Table 12 to 13, the clustering results for randomly generated 2500 (in regression) or 4000 (in classification) data points are presented by cross-tabulation between the clustered index and the true dataset label.

Note that Quad is perfectly covered by Cubic, and Quad and Cubic mainly cover Line in Table 7. Hence, rather than four separate clusters, only two are shown in Fig. 8, each of which implies either sine or polynomial. On the other hand, in Fig. 9, four distinct fine-grained image classification datasets are discriminated by separate clusters.

TABLE 14. Feature extractor g(·) architecture.

Gaussian Process Sine&Polynomial	mini-ImageNet tiered-ImageNet	multi-dataset	
		2 Conv	4 Conv
$\text{Linear}(1, 128)$	$\text{Dropout}(p)$	Conv(32, 5, 1, LRN, MAX)	Conv(32, 3, 1, BN, MAX)
ReLU	$\text{Linear}(640, 128)$	Conv(32, 5, 1, LRN, MAX)	Conv(32, 3, 1, BN, MAX)
$\text{Linear}(128, 128)$	ReLU	$\text{Linear}(32 \times 21 \times 21, 384)$	Conv(32, 3, 1, BN, MAX)
	$\text{Linear}(128, 128)$	ReLU	Conv(32, 3, 1, BN, MAX)
		$\text{Linear}(384, 128)$	$\text{Dropout}(p)$
			$\text{Linear}(32 \times 5 \times 5, 128)$
			ReLU
			$\text{Linear}(128, 128)$

TABLE 15. Regression architecture.

r Encoder	z Encoder	Decoder
$\text{ISAB}_{32}(128+1, 128)$	$\text{ISAB}_{32}(128+1, 128)$	$\text{Linear}(128, 128)$
$\text{ISAB}_{32}(128, 128)$	$\text{ISAB}_{32}(128, 128)$	ReLU
$\text{PMA}_{1}(128, 128)$	$\text{PMA}_{1}(128, 128)$	$\text{Linear}(128, 128)$
$\text{SAB}(128, 128)$	$\text{SAB}(128, 128)$	
$\text{SAB}(128, 256)$	$\text{SAB}(128, 512)$	

TABLE 16. Classification architecture.

r Encoder	z Encoder	Decoder
$\text{ISAB}_{256}(128, 128)$	$\text{ISAB}_{256}(128, 128)$	$\text{Linear}(128, 128)$
$\text{ISAB}_{256}(128, 128)$	$\text{ISAB}_{256}(128, 128)$	ReLU
$\text{PMA}_{1}(128, 128)$	$\text{PMA}_{1}(128, 128)$	$\text{Linear}(128, 128)$
$\text{SAB}(128, 128)$	$\text{SAB}(128, 128)$	
$\text{SAB}(128, 128)$	$\text{SAB}(128, 256)$	

TABLE 17. Hyperparameters for regression.

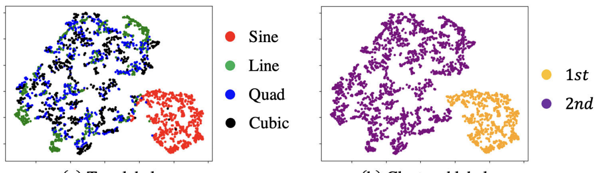
	Gaussian Process	Sine&Polynomial			
		5-*shot*		10-*shot*	
epoch (pre)	-	1e+6		1e+6	
lr (pre)	-	1e-4		1e-4	
β_1	-	1		1	
		Cluster index			
		1st	2nd	1st	2nd
epoch	1e+6	1e+6	1e+6	1e+6	1e+6
lr	1e-4	1e-4	1e-4	1e-4	1e-4
β_2	1	1	1	1	1
β_3	-	0.120	0.941	0.156	0.224

APPENDIX C
ARCHITECTURE DESIGN

We show the detailed architectures used for the feature extractor in Table 14. Here, Conv(d, k, s, n, p) is a convolutional block with d output channels, k kernel size, s stride size, n normalization, and p pooling method. LRN and BN indicate a local response normalization and a batch normalization, respectively, and MAX is a max-pooling with a kernel size of 3 and stride 2. By default, two linear layers are commonly exploited, which come after the convolutional layers if the model input is a 3D image. For the convolutional layers,

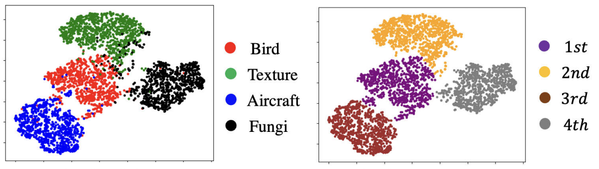
TABLE 18. Hyperparameters for classification.

	mini-ImgeNet		tiered-ImgeNet		multi-dataset							
	1-*shot*	5-*shot*	1-*shot*	5-*shot*	1-*shot*				5-*shot*			
epoch (pre)	-	-	-	-	1.4e+4				1.4e+4			
lr (pre)	-	-	-	-	4.1e-5				4.3e-5			
β_1	-	-	-	-	0.017				0.016			
					Cluster index							
					1st	2nd	3rd	4th	1st	2nd	3rd	4th
epoch	5e+4	6e+4	6e+4	7e+4	2e+4	1.5e+4	5e+3	2e+4	5e+3	1e+4	1e+4	1e+4
lr	8.7e-5	9.2e-5	8.8e-5	9.9e-5	4.4e-5	3.0e-5	7.8e-5	2.5e-5	8.1e-5	7.1e-5	2.9e-5	7.9e-5
p	0.3	0.3	0.47	0.41	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
β_2	0.098	0.004	0.091	0.001	0.063	0.099	0.064	0.020	0.032	0.050	0.032	0.075
β_3	-	-	-	-	0.085	0.068	0.063	0.051	0.018	0.085	0.028	0.053



(a) True label (b) Clustered label

FIGURE 8. t-SNE visualization from sine and polynomial.



(a) True label (b) Clustered label

FIGURE 9. t-SNE visualization from multi-dataset.

we follow the exact setting of [20] depending on whether the model is for task clustering or prediction. For the dropout rate p, please refer to Section VIII.

In Table 15 and 16, the encoder-decoder pipeline of MAHA is summarized. Note that the encoders for r and z are almost the same except for the output size, which is doubled in z due to reparameterization. Also, note that the size of the inputs is different between regression and classification. This is because $g(X)$ and Y are concatenated in regression while *shot*s of $g(X)$ are first divided along *way* by Y and then separately feed-forwarded in classification. Lastly, in regression, the encoder outputs, r and (reparameterized) z, are reshaped from $[batch, 1, 256]$ into $[batch, 2, 128]$ before feed-forwarded into the decoder. By default, all networks use the Adam optimizer with a constant learning rate and an l2 regularization of weight 1e-4.

APPENDIX D
HYPERPARAMETER

Hyperparameters are optimized with the meta-valid set. Among the many hyperparameter optimization processes [77], [78], we use the random search whose outcomes are summarized in Table 17 and 18.

ACKNOWLEDGMENT

Kyeongryeol Go was with the Graduate School of AI, Korea Advanced Institute of Science and Technology (KAIST). The authors declare that there is no conflict of interest regarding the publication of this article.

REFERENCES

[1] L. Brigato and L. Iocchi, “A close look at deep learning with small data,” in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 2490–2497.

[2] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, “A survey on federated learning systems: Vision, hype and reality for data privacy and protection,” *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 2, 2021, doi: 10.1109/TKDE.2021.3124599

[3] K. Choi, J. Yi, C. Park, and S. Yoon, “Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines,” *IEEE Access*, vol. 9, pp. 120043–120065, 2021.

[4] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[5] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1126–113.

[6] M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. W. Teh, D. Rezende, and S. A. Eslami, “Conditional neural processes,” in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1704–1713.

[7] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. Ali Eslami, and Y. Whye Teh, “Neural processes,” 2018, *arXiv:1807.01622*.

[8] H. Kim, A. Mnih, J. Schwarz, M. Garnelo, A. Eslami, D. Rosenbaum, O. Vinyals, and Y. W. Teh, “Attentive neural processes,” in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–18.

[9] M. Kim, K. R. Go, and S.-Y. Yun, “Neural processes with stochastic attention: Paying more attention to the context dataset,” in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–42.

[10] Q. Wang and H. V. Hoof, “Model-based meta reinforcement learning using graph structured surrogate models and amortized policy search,” in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 23055–23077.

[11] A. Pondaven, M. Bakler, D. Guo, H. Hashim, M. Ignatov, and H. Zhu, “Convolutional neural processes for inpainting satellite images,” 2022, *arXiv:2205.12407*.

[12] M. Volpp, F. Flürenbrock, L. Grossberger, C. Daniel, and G. Neumann, “Bayesian context aggregation for neural processes,” in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–29.

[13] T. Nguyen and A. Grover, “Transformer neural processes: Uncertainty-aware meta learning via sequence modeling,” in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 16569–16594.

[14] A. Y. K. Foong, W. P. Bruinsma, J. Gordon, Y. Dubois, J. Requeima, and R. E. Turner, “Meta-learning stationary stochastic process prediction with convolutional neural processes,” 2020, *arXiv:2007.01332*.

[15] S. Markou, J. Requeima, W. P. Bruinsma, A. Vaughan, and R. E. Turner, “Practical conditional neural processes via tractable dependent predictions,” 2022, *arXiv:2203.08775*.

[16] J. Lee, Y. Lee, J. Kim, E. Yang, S. Ju Hwang, and Y. Whye Teh, “Bootstrapping neural processes,” 2020, *arXiv:2008.02956*.

[17] D. Kim, S. Cho, W. Lee, and S. Hong, "Multi-task neural processes," in *Proc. Int. Conf. Learn. Represent.*, 2022, pp. 1–18.
[18] D. Wu, M. Chinazzi, A. Vespignani, Y.-A. Ma, and R. Yu, "Multi-fidelity hierarchical neural processes," 2022, *arXiv:2206.04872*.
[19] R. Vuorio, S.-H. Sun, H. Hu, and J. J. Lim, "Toward multimodal model-agnostic meta-learning," 2018, *arXiv:1812.07172*.
[20] H. Yao, Y. Wei, J. Huang, and Z. Li, "Hierarchically structured meta-learning," 2019, *arXiv:1905.05301*.
[21] H. Yao, X. Wu, Z. Tao, Y. Li, B. Ding, R. Li, and Z. Li, "Automated relational meta-learning," 2020, *arXiv:2001.00745*.
[22] Y. Qian, Y. Zhang, Y. Ye, and C. Zhang, "Distilling meta knowledge on heterogeneous graph for illicit drug trafficker detection on social media," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1–13.
[23] C. Finn, K. Xu, and S. Levine, "Probabilistic model-agnostic meta-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 9516–9527.
[24] A. A. Rusu, D. Rao, J. Sygnowski, O. Vinyals, R. Pascanu, S. Osindero, and R. Hadsell, "Meta-learning with latent embedding optimization," 2018, *arXiv:1807.05960*.
[25] J. Rothfuss, D. Heyn, J. Chen, and A. Krause, "Meta-learning reliable priors in the function space," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1–14.
[26] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," 2016, *arXiv:1606.04080*.
[27] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1199–1208.
[28] B. N. Oreshkin, P. Rodriguez, and A. Lacoste, "TADAM: Task dependent adaptive metric for improved few-shot learning," 2018, *arXiv:1805.10123*.
[29] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1842–1850.
[30] F. Alet, T. Lozano-Pérez, and L. P. Kaelbling, "Modular meta-learning," in *Proc. 2nd Conf. Robot Learn.*, 2018, pp. 856–868.
[31] M. Ren, R. Liao, E. Fetaya, and R. Zemel, "Incremental few-shot learning with attention attractor networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–11.
[32] L. Zintgraf, K. Shiarli, V. Kurin, K. Hofmann, and S. Whiteson, "Fast context adaptation via meta-learning," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 7693–7702.
[33] X. Zhang, D. Meng, H. Gouk, and T. Hospedales, "Shallow Bayesian meta learning for real-world few-shot recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 651–660.
[34] Q. Zhang, J. Fang, Z. Meng, S. Liang, and E. Yilmaz, "Variational continual Bayesian meta-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 1–13.
[35] R. Ni, M. Goldblum, A. Sharaf, K. Kong, and T. Goldstein, "Data augmentation for meta-learning," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 8152–8161.
[36] J. Oh, H. Yoo, C. Kim, and S.-Y. Yun, "BOIL: Towards representation change for few-shot learning," 2020, *arXiv:2008.08882*.
[37] C. K. I. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2, no. 3. Cambridge, MA, USA: MIT Press, 2006.
[38] A. G. Wilson, Z. Hu, R. Salakhutdinov, and E. P. Xing, "Deep kernel learning," in *Proc. 19th Int. Conf. Artif. Intell. Statist.*, 2016, pp. 370–378.
[39] P. Tossou, B. Dura, F. Laviolette, M. Marchand, and A. Lacoste, "Adaptive deep kernel learning," 2019, *arXiv:1905.12131*.
[40] S. W. Ober, C. E. Rasmussen, and M. V. D. Wilk, "The promises and pitfalls of deep kernel learning," in *Proc. Conf. Uncertainty Artif. Intell.*, 2021, pp. 1206–1216.
[41] J. Requeima, J. Gordon, J. Bronskill, S. Nowozin, and R. E. Turner, "Fast and flexible multi-task classification using conditional neural adaptive processes," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 7959–7970.
[42] T. A. Le, H. Kim, M. Garnelo, D. Rosenbaum, J. Schwarz, and Y. W. Teh, "Empirical evaluation of neural process objectives," in *Proc. NeurIPS Workshop Bayesian Deep Learn.*, 2018.
[43] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.
[44] Z. Jiang, Y. Zheng, H. Tan, B. Tang, and H. Zhou, "Variational deep embedding: An unsupervised and generative approach to clustering," 2016, *arXiv:1611.05148*.
[45] W. Joo, W. Lee, S. Park, and I.-C. Moon, "Dirichlet variational autoencoder," *Pattern Recognit.*, vol. 107, Nov. 2020, Art. no. 107514.
[46] E. Nalisnick and P. Smyth, "Stick-breaking variational autoencoders," 2016, *arXiv:1605.06197*.
[47] N. Li, W. Li, Y. Jiang, and S.-T. Xia, "Deep Dirichlet process mixture models," in *Proc. 37th Conf. Uncertainty Artif. Intell.*, 2022, pp. 1138–1147.
[48] G. Denevi, M. Pontil, and C. Ciliberto, "Conditional meta-learning of linear representations," 2021, *arXiv:2103.16277*.
[49] M. K. Titsias, F. J. Ruiz, S. Nikoloutsopoulos, and A. Galashov, "Information theoretic meta learning with Gaussian processes," in *Proc. Conf. Uncertainty Artif. Intell.*, 2021, pp. 1597–1606.
[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[51] S. Zhao, J. Song, and S. Ermon, "Towards deeper understanding of variational autoencoding models," 2017, *arXiv:1702.08658*.
[52] H. Fu, C. Li, X. Liu, J. Gao, A. Celikyilmaz, and L. Carin, "Cyclical annealing schedule: A simple approach to mitigating KL vanishing," 2019, *arXiv:1903.10145*.
[53] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improving variational inference with inverse autoregressive flow," 2016, *arXiv:1606.04934*.
[54] J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh, "Set transformer: A framework for attention-based permutation-invariant neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 3744–3753.
[55] C. Cremer, X. Li, and D. Duvenaud, "Inference suboptimality in variational autoencoders," 2018, *arXiv:1801.03558*.
[56] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," 2016, *arXiv:1601.06759*.
[57] S. Zhao, J. Song, and S. Ermon, "Learning hierarchical features from generative models," 2017, *arXiv:1702.08396*.
[58] J. Lei Ba, J. Ryan Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.
[59] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 3483–3491.
[60] Y. W. Teh, V. Bapst, W. M. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu, "Distral: Robust multitask reinforcement learning," 2017, *arXiv:1707.04175*.
[61] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks via self distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3713–3722.
[62] H. Lee, S. J. Hwang, and J. Shin, "Self-supervised label augmentation via input transformations," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 5714–5724.
[63] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "Beta-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. ICLR*, 2016, pp. 1–22.
[64] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel, "Meta-learning for semi-supervised few-shot classification," 2018, *arXiv:1803.00676*.
[65] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *Proc. ICLR*, 2016, pp. 1–11.
[66] S. Qiao, C. Liu, W. Shen, and A. Yuille, "Few-shot image recognition by predicting parameters from activations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7229–7238.
[67] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *J. Mach. Learn. Res.*, vol. 6, pp. 1939–1959, Dec. 2005.
[68] M. Yin, G. Tucker, M. Zhou, S. Levine, and C. Finn, "Meta-learning without memorization," 2019, *arXiv:1912.03820*.
[69] L. V. D. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–27, 2008.
[70] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "A critical analysis of self-supervision, or what we can learn from a single image," 2019, *arXiv:1904.13132*.
[71] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[72] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD birds 200," Caltech, Tech. Rep. CNS-TR-201, Jan. 2010. [Online]. Available: http://www.vision.caltech.edu/visipedia/CUB-/se3/wp-content/uploads/2014/09/WelinderEtal10_CUB-200.pdf

[73] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*. Princeton, NJ, USA: Citeseer, 2014, pp. 3606–3613.

[74] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," 2013, *arXiv:1306.5151*.

[75] Y. C. Beejisbrigit, "FGCVx fungi classification challenge," Kaggle, 2018. [Online]. Available: https://kaggle.com/competitions/fungi-challenge-fgvc-2018

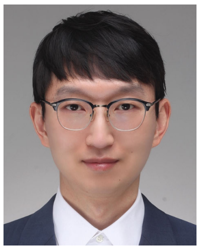
[76] P. Virtanen et al., "SciPy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, pp. 261–272, Feb. 2020.

[77] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. 2, pp. 1–25, 2012.

[78] J. T. Wilson, F. Hutter, and M. Peter Deisenroth, "Maximizing acquisition functions for Bayesian optimization," 2018, *arXiv:1805.10196*.

KYEONGRYEOL GO received the B.S. degree in industrial and systems engineering and the M.S. degree from the Graduate School of AI, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2019 and 2021, respectively, under the supervision of Prof. Seyoung Yun. In 2019, he interned at the Data Science Department, SK Hynix, Icheon, South Korea. In 2021, he was involved in designing a deep reinforcement learning agent for architectural design and crop cultivation at Spacewalk, Seoul, South Korea. He is a Machine Learning Engineer with Superb AI, Seoul. His research interests include stochastic process, variational inference, optimization, and deriving reliable predictions under uncertainties.

MINGYU KIM received the B.S. degree in industrial engineering from Konkuk University, South Korea, in 2012, and the M.S. degree in ocean system engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2014, where he is currently pursuing the Ph.D. degree with the Graduate School of AI. He was a Research Engineer at Samsung Heavy Industries, South Korea, from 2014 to 2019. His research interests include probabilistic models, variational inference, generative models, Bayesian deep learning, implicit representation, and meta-learning.



SEYOUNG YUN received the B.S. and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2006 and 2012, respectively, under the supervision of Prof. Dong-Ho Cho. From 2012 to 2013, he was a Postdoctoral Researcher at KAIST (hosted by Prof. Yung Yi). He was a Visiting Researcher at Microsoft Research, Cambridge, U.K. (hosted by Prof. Milan Vojnovic). He was a Postdoctoral Researcher at the Los Alamos National Laboratory, NM, USA (hosted by Dr. Michael Chertkov), the Microsoft Research-INRIA Joint Center, Paris, France (hosted by Dr. Marc Lelarge), and KTH, Stockholm, Sweden (hosted by Prof. Alexandre Proutiere). He is an Associate Professor with the Graduate School of AI, KAIST. He received the Best Paper Award from ACM MOBIHOC, in 2013, and the NIPS Outstanding Reviewer Award, in 2016.