# Learning to Learn with Contrastive Meta-Objective

**Anonymous Author(s)**
Affiliation
Address
email

## Abstract

We propose a contrastive meta-objective to enable meta-learners to emulate human-like rapid learning capability through enhanced alignment and discrimination. Our proposed approach, dubbed ConML, exploits task identity as additional supervision signal for meta-training, benefiting meta-learner's fast-adaptation and task-level generalization abilities. This is achieved by contrasting the outputs of meta-learner, i.e, performing contrastive learning in the model space. Specifically, we introduce metrics to minimize the inner-task distance, i.e., the distance among models learned on varying data subsets of the same task, while maximizing the inter-task distance among models derived from distinct tasks. ConML distinguishes itself through versatility and efficiency, seamlessly integrating with episodic meta-training methods and the in-context learning of large language models (LLMs). We apply ConML to representative meta-learning algorithms spanning optimization-, metric-, and amortization-based approaches, and show that ConML can universally and significantly improve conventional meta-learning and in-context learning.

## 1 Introduction

Meta-learning [37, 42], or learning to learn, is a powerful paradigm that aims to enable a learning system to quickly adapt to new tasks. Meta-learning has been widely applied in different fields, like few-shot learning [17, 50], reinforcement learning [56, 26] and neural architecture search [16, 38]. In meta-training, a meta-leaner mimics the learning processes on many relevant tasks to gain experience about how to make adaptation. In meta-testing, the meta-trained adaptation process is performed on unseen tasks. The adaptation process is achieved by generating task-specific model by the meta-learner, which is given a set of training examples and returns a predictive model. People prefer meta-learning to equip models with human's fast learning ability, so that a good model can be achieved with a few examples [50].

The combination of two cognitive capabilities, namely, **alignment** and **discrimination**, is essential for human's fast learning ability [23, 12, 13]. A good learner possesses the alignment [27] ability to align different partial views of a certain object, which means they can integrate various aspects or perspectives of information to form a coherent understanding. On the other hand, discrimination [34] refers to the learner's capacity to distinguish between one stimulus and similar stimuli, responding appropriately only to the correct stimuli. This is a fundamental ability that allows learners to differentiate between what is relevant and what is not, ensuring that their responses are accurate and based on the correct understanding of the stimuli presented. With alignment and discrimination, learners can synthesize fragmented information to construct a complete picture of an object or concept, while also being able to discern subtle differences between distinct but similar objects or ideas. Such learners are not only efficient in processing information but also in applying their knowledge accurately in varied contexts. This dual capability is crucial for effective learning.

We expect meta-learners to emulate the above combination of alignment and discrimination capabilities to approach human's fast learning ability. By equipping a meta-learner with the ability to

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

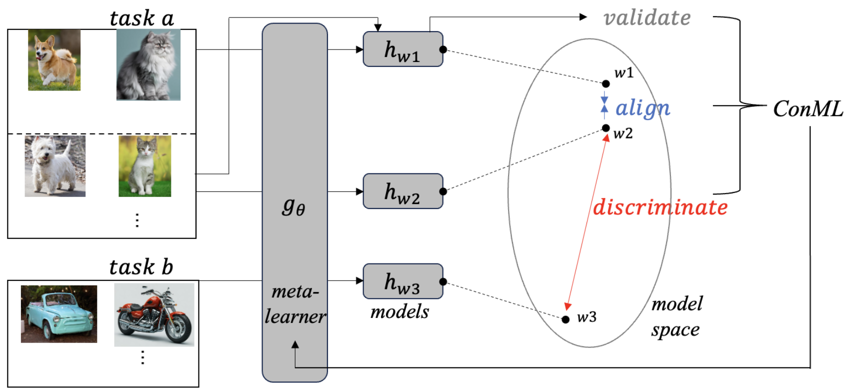



Figure 1: ConML is performing contrastive learning in model space, where alignment and discrimination encourage the meta-learner's fast-adaptation and task-level generalize ability respectively.

align, we enable it to capture the core essence of a task and being invariant to noises. Meanwhile, discrimination ensures that a meta-learner can learn specific models for unique tasks, as it is a natural supposition that different tasks enjoy distinguishable models. This reflects the natural diversity of problems we encounter in the real world and the varied strategies we employ to solve them. Together, alignment and discrimination empower a meta-learner to not only grasp the subtleties of individual tasks but also to generalize its learning across a spectrum of challenges. This dual capability can makes a meta-learner robust, versatile, and more aligned with the nuanced nature of human learning and reasoning. However, existing meta-learning approaches conventionally follows the idea of "train as you test", to minimize the validation loss [46] of meta-training tasks as meta-objective, where supervision signal are directly produced by sample labels. To provide stronger supervision, there are works assuming that the task-specific target models of meta-training tasks are available, then the meta-training can be supervised by aligning the learned model and the corresponding target model, with model weights [51, 52] or knowledge distillation [55]. However, as the target models are expensive to learn, and even not available in many real world problems, meta-objectives requiring the target models have very restricted applications. Moreover, the importance of discrimination ability of meta-learner has not been noticed in the literature.

To achieve this, we propose contrastive meta-learning (ConML), by directly contrasting the outputs of meta-learner in the model space, shown in Figure 1. Conventional contrastive learning (CL) [14, 48, 44] learns an encoder in unsupervised manner by equipping the model with alignment and discrimination ability by exploiting the distinguishable identity of unlabeled samples. Considering tasks in meta-learning are also unlabeled but have distinguishable identity, we are inspired to adopt similar strategy in meta-learning. ConML exploits tasks as CL exploits unlabeled samples. Positive pairs in ConML are different subsets of the same task, while negative pairs are datasets of different tasks. In the model space output by meta-learner, inner-task distance can be measured between positive pairs and inter-task distance can be measured between negative pairs. The contrastive meta-objective is minimizing inner-task distance while maximizing inter-task distance, corresponding to the expected alignment and discrimination ability respectively. The proposed ConML is universal and cheap, as it can be plugged-in any meta-learning algorithms following the episodic training, and does not require additional data nor model training. In this paper, we widely study ConML on representative meta-learning algorithms from different categories: optimization-based (e.g., MAML [17]), metric-based (e.g., ProtoNet [39]), amortization-based (e.g., Simple CNAPS [6]). We also investigate in-context learning [8] with reformulating it into the meta-learning paradigm, and show how ConML integrates and helps.

Our contributions are:

- We propose to emulate cognitive alignment and discrimination capabilities in meta-learning, to narrow down the gap of fast learning ability between meta-learners and humans.
- We generalize contrastive learning from representation space of unsupervised learning to model space of meta-learning. The exploiting task identity as additional supervision benefits meta-learner's fast-adaptation and task-level generalize abilities.
- ConML is algorithm-agnostic, that can be incorporated into any meta-learning algorithms with episodic training. We empirically show ConML can bring universal improvement with cheap implementation on a wide range of meta-learning algorithms and in-context learning.

## 2 Related Works

### 2.1 Learning to Learn

Meta-learning learns to improve the learning algorithm itself [37], i.e., learns to learn. Popular meta-learning approaches can be roughly divided into three categories [7]: optimization-based, metric-based and amortization-based. Optimization-based approaches [4, 17, 28] focus on learning better optimization strategies for adapting to new tasks. For example MAML [17] learns initial model parameters, where few steps of gradient descent can quickly make adaptaion for specific tasks. Metric-based approaches [46, 39, 41] leverages learned similarity metrics. For example, Prototypical Networks [39] and Matching Networks [46] learn global shared encoders to map training set to embeddings, based on which task-specific model can be built. Amortization-based approaches [19, 33, 6] seek to learn a shared representation across tasks. They amortize the adaptation process by using neural networks to directly infer task-specific parameters from training set. Examples are CNPs [19] and CNAPs [33].

In-context learning (ICL) [8] is designed for large language models, which integrates examples (input-output pairs) in a task and a query input into the prompt, thus the language model can answer the query. Recently, ICL has been studied as a general approach of learning to learn [2, 18, 47, 1], which reduces meta-learning to conventional supervised learning via training a sequence model. It considers training set as context to be provided along with the input to predict, forming a sequence to feed the model. Training such a model can be viewed as an instance of meta-learning [18].

### 2.2 Contrastive Learning

Contrastive learning is a powerful technique in representation learning [29, 10, 48]. Its primary goal is to learn useful representations, which are invariant to unnecessary details, and preserve as much information as possible. This is achieved by maximizing alignment and discrimination (uniformity) in representation space [48]. In conventional contrastive learning, alignment refers to bringing positive pairs (e.g., augmentations of the same sample [54, 22, 5, 21, 10]) closer together in the learned representation space. By maximizing alignment, the representations are encouraged to be invariant to unneeded noise factors. Discrimination refers to separating negative pairs (e.g., different samples) farther. Maximizing discrimination without any other knowledge results in uniformity, i.e., uniform distribution in the representation space. By maximizing discrimination, the representations are encouraged to preserve as much information of the data as possible [43, 5], benefiting the generalization ability.

## 3 Meta-Learning with Contrastive Meta-Objective

Meta-learning is a methodology considered with "learning to learn" machine learning algorithms. Define $\mathcal{L}(\mathcal{D}; h)$ as the loss obtained by evaluating model $h$ on dataset $\mathcal{D}$ with function $\ell(y, \hat{y})$ (e.g., cross entropy or mean squared loss), $g(;\theta)$ is a meta-learner that maps a dataset $\mathcal{D}$ to a model $h$, i.e, $h = g(\mathcal{D}; \theta)$. Given a distribution of tasks $p(\tau)$, where each task $\tau$ consists of a training set $\mathcal{D}^{\text{tr}}_\tau = \{(x_{\tau,i}, y_{\tau,i})\}_{i=1}^n$, and a validation set $\mathcal{D}^{\text{val}}_\tau = \{(x_{\tau,i}, y_{\tau,i})\}_{i=n+1}^m$, the goal of meta-learning is to learn $g(;\theta)$ to perform well on new task $\tau'$ sampled from $p(\tau')$, evaluated by $\mathcal{L}(\mathcal{D}^{\text{val}}_{\tau'}; g(\mathcal{D}^{\text{tr}}_{\tau'}; \theta))$.

### 3.1 A Unified View of Episodic Training

We aim to introduce "learning to align and discriminate" to universally improve the meta-learning process. The most conventional way of meta-training is taking the *validation loss* as meta-objective to optimize $\theta$:

$$\min_\theta \mathbb{E}_{\tau\sim p(\tau)} \mathcal{L}(\mathcal{D}^{\text{val}}_\tau; g(\mathcal{D}^{\text{tr}}_\tau; \theta)). \tag{1}$$

Different meta-learning algorithms tailor the function inside $g$, while sharing the same episodic meta-training to achieve (1). Shown as Algorithm 1, in each episode, $B$ tasks are sampled from $p(\tau)$ to form a batch $\mathbf{b}$, and validation loss of each task is aggregated as the supervision signal $L_v = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} \mathcal{L}(\mathcal{D}^{\text{val}}_\tau; g(\mathcal{D}^{\text{tr}}_\tau; \theta))$ to update $\theta$. By specifying the function inside $g$, Algorithm 1 can generalize the meta-training process of different meta-learning algorithms.

**Algorithm 1** Mini-Batch Episodic Meta-Training (Conventional)

```
while Not converged do
    Sample a batch of tasks b ~ p^B(τ).
    for All τ ∈ b do
        Get task-specific model h_τ = g(D^tr_τ; θ);
        Get validation loss L(D^val_τ; h_τ);
    end for
    L_v = 1/B Σ_{τ∈b} L(D^val_τ; g(D^tr_τ; θ))
    Update θ by θ ← θ − ∇_θ L_v.
end while
```

Table 1: Specifications of ConML.

| Category | Examples | $g(\mathcal{D};\theta)$ | $\psi(g(\mathcal{D};\theta))$ |
|---|---|---|---|
| Optimization -based | MAML[17], Reptile[28] | Update model weights $\theta - \nabla_\theta \mathcal{L}(\mathcal{D}; h_\theta)$ | $\theta - \nabla_\theta \mathcal{L}(\mathcal{D}; h_\theta)$ |
| Metric -based | ProtoNet[39], MatchNet[46] | Build classifier with $\{(\{f_\theta(x_i)\}_{x_i \in \mathcal{D}_j}, j)\}_{j=1}^N$ | Concatenate $[\frac{1}{\lvert\mathcal{D}_j\rvert} \sum_{x_i \in \mathcal{D}_j} f_\theta(x_i)]_{j=1}^N$ |
| Amortization -based | CNPs[19], CNAPs[33] | Map $\mathcal{D}$ to model weights by $H_\theta(\mathcal{D})$ | $H_\theta(\mathcal{D})$ |

Specifications of optimization-based, metric-based and amortization-based algorithms are summarized in Table 1.

We design ConML to be integrated with Algorithm 1 without specifying $g$, thus to be universally applicable for meta-learning algorithms following the episodic manner. In Section 3.2, we introduce how to measure the objective. Then in Section 3.3, we introduce specifications of ConML on a wide range of meta-learning algorithms.

### 3.2 Integration with Episodic Meta-Training

To equip meta-learners with the desired alignment and discrimination ability, we design contrastive meta-objective measured in the output space of meta-learner, i.e., the model space of $h$. Alignment is achieved by minimizing inner-task distance, which is the distance among models generated from different subsets of the same task. Discrimination is achieved by maximize the inter-task distance, which is the distance among models generated from different tasks. Here we introduce how to measure the contrastive objective and perform optimization.

**Obtaining Model Representation.** To train the meta-learner $g$, the distances $D^{\text{in}}, D^{\text{out}}$ are measured in the output space of $g$, i.e., the model space $\mathcal{H}$. A feasible way is to first represent model $h = g(\mathcal{D};\theta) \in \mathcal{H}$ as fixed length vectors $\boldsymbol{e} \in \mathbb{R}^d$, then measure by explicit distance function $\phi(\cdot,\cdot)$ (e.g., cosine distance). Note that $\mathcal{H}$ is algorithm-specific. Here we only introduce a projection $\psi : \mathcal{H} \to \mathbb{R}^d$ to obtain model representations $\boldsymbol{e} = \psi(h)$. The $\mathcal{H}$ and $\psi$ will be elucidated and specified for different meta-learning algorithms in Section 3.3.

**Obtaining Inner-Task Distance.** During meta-training, $\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}$ contains all the available information about task $\tau$. The meta-learner is expected to learn similar model given any subset $\kappa$ of the task. Meanwhile those models from subsets are expected to be similar to the model learned from the full supervision $\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}$. We design the following inner-task distance to minimize that encourages $g$ to learn a generalizable model even from a set containing only few or biased samples. For $\forall \kappa \subseteq \mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}$, we expect $\boldsymbol{e}_\tau^\kappa = \boldsymbol{e}_\tau^*$, where $\boldsymbol{e}_\tau^\kappa = \psi(g(\kappa;\theta)), \boldsymbol{e}_\tau^* = \psi(g(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}};\theta))$. The inner-task distance $D_\tau^{\text{in}}$ of task $\tau$ is defined as:

$$D_\tau^{\text{in}} = \frac{1}{K} \sum\nolimits_{k=1}^{K} \phi(\boldsymbol{e}_\tau^{\kappa_k}, \boldsymbol{e}_\tau^*), \ s.t., \boldsymbol{e}_\tau^{\kappa_k} \sim \pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}), \tag{2}$$

where $\{\kappa_k\}_{k=1}^K$ are $K$ subsets sampled from $\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}$ by certain sampling strategy $\pi_\kappa$. In each episode given a batch of task $\mathbf{b}$ containing $B$ tasks, inner-task distance is averaged by $D^{\text{in}} = \frac{1}{B} \sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$.

**Obtaining Inter-Task Distance.** Since the goal of meta-learning is improving the performance on unseen tasks, it is important that the $g$ is generalizable for diverse tasks. With a natural supposition that different tasks enjoy different task-specific models, it is necessary that $g$ can learn different models from different tasks, i.e., discrimination. We define the following inter-task distance to maximize to improve the task-level generalizability of $g$. For two tasks $\tau \neq \tau'$ during meta-training, we expect to maximize the distance between $\boldsymbol{e}_\tau^*$ and $\boldsymbol{e}_{\tau'}^*$. To be practical under the mini-batch episodic training paradigm, we consider to measure inter-task distance among a batch of tasks:

$$D^{\text{out}} = \frac{1}{B(B-1)} \sum\nolimits_{\tau \in \boldsymbol{b}} \sum\nolimits_{\tau' \in \boldsymbol{b} \setminus \tau} \phi(\boldsymbol{e}_\tau^*, \boldsymbol{e}_{\tau'}^*). \tag{3}$$

**Training Procedure.** ConML measures $D^{\text{in}}$ by (2) and $D^{\text{out}}$ by (3) in each episode, and minimizes a combination of the validation loss $L_v$ and contrastive meta-objective $D^{\text{in}} - D^{\text{out}}$:

$$L = L_v + \lambda(D^{\text{in}} - D^{\text{out}}). \quad (4)$$

The training procedure of ConML is provided in Algorithm 2. Comparing with Algorithm 1, ConML introduces additional computation $\psi(g(\mathcal{D};\theta))$ for $K+1$ times in each episode. Note that we implement $\psi$ with very cheap function such as obtaining model weights (or a single probing, i.e., feeding-forward, for ICL), and $g(\mathcal{D};\theta)$ already exists in Algorithm 1 while multiple $g(\mathcal{D};\theta)$ can be parallel-computed. ConML could have very comparable time consumption.

**Algorithm 2** Meta-Learning with Contrastive Meta-Object (ConML)

```
while Not converged do
    Sample a batch of tasks b ~ p^B(τ).
    for All τ ∈ b do
        for k = 1, 2, ···, K do
            Sample κ_k from π_κ(D_τ^tr ∪ D_τ^val);
            Get model representation e_τ^{κ_k} = ψ(g(κ_k; θ));
        end for
        Get model representation e_τ^* = ψ(g(D_τ^tr ∪ D_τ^val; θ));
        Get inner-task distance D_τ^in by (2);
        Get task-specific model h_τ = g(D_τ^tr; θ);
        Get validation loss L(D_τ^val; h_τ);
    end for
    Get D^in = (1/B) Σ_{τ∈b} D_τ^in and D^out by (3);
    Get loss L by (4);
    Update θ by θ ← θ − ∇_θ L.
end while
```

### 3.3 Instantiations of ConML

Here we demonstrate specifications of $\mathcal{H}$ and $\psi(g(\mathcal{D},\theta))$ to obtain model representation to implement ConML. We show examples on representative meta-learning algorithms from different categories: optimization-based, metric-based and amortization-based. They are explicitly represented by model weights, summarized in Table 1.

**With Optimization-Based Methods.** The representative algorithm of optimization-based meta-learning is MAML. It meta-learns an initialization from where gradient steps are taken to learn task-specific models, i.e., $g(\mathcal{D};\theta) = h_{\theta-\nabla_\theta \mathcal{L}(\mathcal{D};h_\theta)}$. As $g$ directly generates the model weights, we explicitly take the model weights as model representation. The representation of model learned by $g$ given a dataset $\mathcal{D}$ is $\psi(g(\mathcal{D};\theta)) = \theta - \nabla_\theta \mathcal{L}(\mathcal{D};h_\theta)$. Note that there are optimization-based meta-learning algorithms which are based on first-order approximation of MAML, thus they do not strictly follows Algorithm 1 to minimize validation loss (e.g., FOMAML [17] and Reptile [28]). ConML can also be incorporated as long as it follows the episodic manner.

**With Metric-Based Methods.** Metric-based algorithms are feasible for classification tasks. Given dataset $\mathcal{D}$ of a $N$-way classification task, metric-based algorithms can be summarized as classifying according to distances with $\{\{f_\theta(x_i)\}_{x_i\in\mathcal{D}_j}\}_{j=1}^N$ and corresponding labels, where $f_\theta$ is a meta-learned encoder and $\mathcal{D}_j$ is the set of inputs belongs to class $j$. We design to represent this metric-based classifier with the concatenation of mean embedding of each class in label-aware order. For example, ProtoNet [39] computes the prototype $\boldsymbol{c}_j$ , i.e., mean embedding of samples in each class. $\boldsymbol{c}_j = \frac{1}{|\mathcal{D}_j|}\sum_{(x_i,y_i)\in\mathcal{D}_j} f_\theta(x_i)$. Then classifier $h_{\theta,\mathcal{D}}$ is built by giving prediction $p(y=j \mid x) = \exp(-d(f_\theta(x),\boldsymbol{c}_j))/\sum_{j'}\exp(-d(f_\theta(x),\boldsymbol{c}_{j'}))$. As the outcome model $h_{\theta,\mathcal{D}}$ depends on $\mathcal{D}$ through $\{\boldsymbol{c}_j\}_{j=1}^N$ and corresponding labels, the representation is specified as $\psi(g(\mathcal{D};\theta)) = [\boldsymbol{c}_1|\boldsymbol{c}_2|\cdots|\boldsymbol{c}_N]$, where $[\cdot|\cdot]$ means concatenation.

**With Amortization-Based Methods.** Amortization-based approaches meta-learns a hypernetwork $H_\theta$, which aggregates information from $\mathcal{D}$ to task-specific parameter $\alpha$ and serves as weights of main-network $h$, resulting in task-specific model $h_\alpha$. For example, Simple CNAPS [6] adopts the hypernetwork to generate only a small amount of task-specific parameter, which performs feature-wise linear modulation (FiLM) on convolution channels of the main-network. For contrasting we represent $h_\alpha$ by $\alpha$, i.e., the output of hypernetwork $H_\theta$: $\psi(g(\mathcal{D};\theta)) = H_\theta(\mathcal{D})$. The detailed procedures of different meta-learning algorithms with ConML are provided in Appendix A.

## 4 In-Context Learning with Contrastive Meta-Objective

In-context learning (ICL) is first proposed for large language models [8], where examples in a task are integrated into the prompt (input-output pairs) and given a new query input, the language model can generate the corresponding output. This approach allows pre-trained model to address new tasks without fine-tuning the model. For example, given "*happy->positive; sad->negative; blue->*", the model can output "*negative*", while given "*green->cool; yellow->warm; blue->*" the model can output "*cool*". ICL has the ability to learn from the prompt. Training ICL can be viewed as learning to learn, like meta-learning [25, 18, 24]. More generally, the input and output are not necessarily to be natural language. In ICL, a sequence model $T_\theta$ (typically transformer [45]) is trained to map sequence $[x_1, y_1, x_2, y_2, \cdots, x_{m-1}, y_{m-1}, x_m]$ (prompt prefix) to prediction $y_m$. Given distribution $P$ of training prompt $t$, then training ICL follows an auto-regressive manner:

$$\min_\theta \mathbb{E}_{t\sim P(t)} \frac{1}{m} \sum\nolimits_{i=0}^{m-1} \ell(y_{t,i+1}, T_\theta([x_{t,1}, y_{t,1}, \cdots, x_{t,i+1}])). \tag{5}$$

It has been mentioned that the training of ICL can be viewed as an instance of meta-learning [18, 2] as $T_\theta$ learns to learn from prompt. In this section we first formally reformulate $T_\theta$ to meta-learner $g(;\theta)$, then introduce how ConML can be integrated with ICL.

### 4.1 A Meta-learning Reformulation

Denote a sequentialized $\mathcal{D}$ as $\vec{\mathcal{D}}$ where the sequentializer is default to bridge $p(\tau)$ and $P(t)$. Then the prompt $[x_{\tau,1}, y_{\tau,1}, \cdots, x_{\tau,m}, y_{\tau,m}]$ can be viewed as $\vec{\mathcal{D}}_\tau^{tr}$ which is providing task-specific information. Note that ICL does not specify an explicit output model $h(x) = g(\mathcal{D};\theta)(x)$; instead, this procedure exists only implicitly through the feeding-forward of the sequence model, i.e., task-specific prediction is given by $g([\vec{\mathcal{D}}, x];\theta)$. Thus we can reformulate the training of ICL (5) as:

$$\min_\theta \mathbb{E}_{\tau\sim p(\tau)} \frac{1}{m} \sum\nolimits_{i=0}^{m-1} \ell(y_{\tau,i+1}, g([\vec{\mathcal{D}}_{\tau,0:i}, x_{\tau,i+1}];\theta)). \tag{6}$$

Equation (6) can be regarded as the validation loss (1) in meta-learning, where each task in each episode is sampled multiple times to form $\mathcal{D}_\tau^{\text{val}}$ and $\mathcal{D}_\tau^{\text{tr}}$ in an auto-regressive manner. The training of ICL thus follows the episodic meta-training (Algorithm 1), where the validation loss with determined $\mathcal{D}_\tau^{\text{tr}}$ and $\mathcal{D}_\tau^{\text{val}}$: $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; g(\mathcal{D}_\tau^{\text{tr}};\theta))$, is replaced by loss validated in the auto-regressive manner: $\frac{1}{m}\sum_{i=0}^{m-1} \ell(y_{\tau,i+1}, g([\vec{\mathcal{D}}_{\tau,0:i}, x_{\tau,i+1}];\theta))$.

### 4.2 Integration with ICL

Since the training of ICL could be reformulated as episodic meta-training, the three steps to measure ConML proposed in Section 3.2 can be also adopted for ICL, but the first step to obtain model representation $\psi(g(\mathcal{D},\theta))$ needs modification. Due to the absence of an inner learning procedure for a predictive model for prediction $h(x) = g(\mathcal{D};\theta)(x)$, representation by explicit model weights of $h$ is not feasible for ICL.

To represent what $g$ learns from $\mathcal{D}$, we design to incorporate $\vec{\mathcal{D}}$ with a dummy input $u$, which functions as a probe and its corresponding output can be readout as representation:

$$\psi(g(\mathcal{D};\theta)) = g([\vec{\mathcal{D}}, u];\theta), \tag{7}$$

where $u$ is constrained to be in the same shape as $x$, and has consistent value in an episode. The complete algorithm of ConML for ICL is provided in Appendix A. From the perspective of learning to learn, ConML encourages ICL to align and discriminate like it does for conventional meta-learning, while the representations to evaluate inner- and inter- task distance are obtained by probing output rather than explicit model weights. Thus, incorporating ConML into the training process of ICL benefits the fast-adaptation and task-level generalization ability. From the perspective of supervised learning, ConML is performing unsupervised data augmentation that it introduces the dummy input and contrastive objective as additional supervision to train ICL.

## 5 Experiments

In this secrion, we first empirically investigate the alignment and discrimination empowered by ConML. Then we show the effect of ConML that it significantly improve meta-learning performance on a wide range of meta-learning algorithms on few-shot image classification, and the effect of ConML-ICL with in-context learning general functions. Additionally, by applying ConML we provide a SOTA approach for few-shot molecular property prediction problem, provided in Appendix B. Code is provided in supplementary materials.

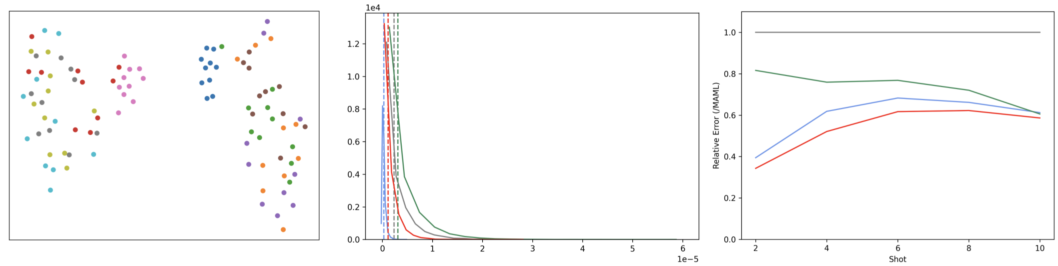
### 5.1 Impact of Alignment and Discrimination

There are two important questions to understand the way ConML works: First, does ConML equip meta-learners with better alignment and discrimination as expected? Second, what is the contribution of inner-task and inter-task distance respectively? We take ConML-MAML as example and investigate above questions with few-shot regression problem following the same settings in [17], where each task involves regressing from the input to the output of a sine wave. We use this synthetic regression

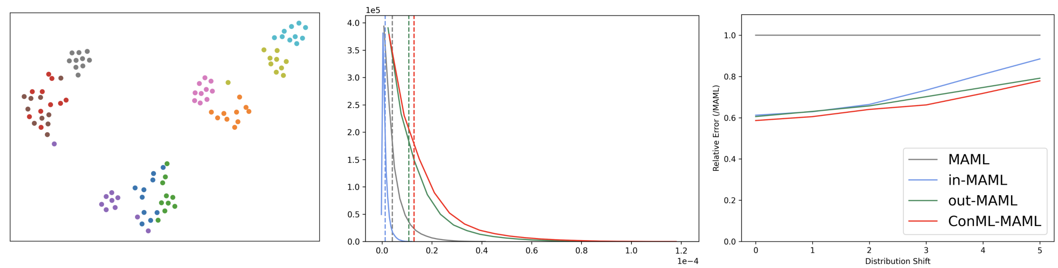
Table 2: Meta-testing and clustering performance of few-shot sinusoidal regression.

| Method | MSE (5-shot) | MSE (10-shot) | Silhouette | DBI | CHI |
|---|---|---|---|---|---|
| MAML | $.6771 \pm .0377$ | $.0678 \pm .0022$ | $.1068 \pm .0596$ | $.0678 \pm .0021$ | $31.55 \pm 2.52$ |
| ConML-MAML | $\mathbf{.3935} \pm .0100$ | $\mathbf{.0397} \pm .0009$ | $\mathbf{.1945} \pm .0621$ | $\mathbf{.0397} \pm .0009$ | $\mathbf{39.22} \pm 2.61$ |

dataset to be able to sample data and vary the distribution as needed for investigation. The implement of ConML-MAML is consistent with Section 5.2. Firstly the meta-testing performance in Table 2 shows that ConML is effective for the regression problem.




(a) Model distribution of MAML. (b) Inner-task distance distribution. (c) Varying test shots.




(d) Model distribution of ConML-MAML. (e) Inter-task distance distribution. (f) Varying test distribution.

Figure 2: Investigating the way ConML works.

**Clustering.** If ConML enhances the alignment and discrimination abilities, ConML-MAML can generate more similar models from different subsets of the same task, while generating more separable models from different tasks. This can be verified by evaluating the clustering performance for model representations $\boldsymbol{e}$. During meta-testing, we randomly sample 10 different tasks, inside each we sample 10 different subsets, each one contains $N = 10$ samples. Taking these 100 different $\mathcal{D}^{\text{tr}}$ as input, meta-learner generates 100 models. Figure 2(a) and 2(d) show the visualization of model distribution. It can be obviously observed ConML-MAML performs better alignment and discrimination than MAML. To quantity the results, we also evaluate the supervised clustering performance, where task identity is used as label. Table 2 shows the supervised clustering performance of different metrics: Silhouette score [35], Davies-Bouldin index (DBI) [15] and Calinski-Harabasz index (CHI) [9], where ConML-MAML shows much better performance.

**Decoupling Inner- and Inter-Task Distance.** In conventional unsupervised contrastive learning, where objective only relies on contrasting of positive pairs and negative pairs, positive and negative pairs are both necessary to avoid learning representations without useful information. However, in ConML, there is validation loss $L_v$ plays a necessary and fundamental role in "learning to learn", and the contrastive objective is introduced as additional supervision to enhance alignment and discrimination. Thus, distance of positive pairs ($D^{\text{in}}$) and negative pairs ($D^{\text{out}}$) in ConML could be decoupled and incorporated with $L_v$ respectively. We aim to understand how $D^{\text{in}}$ and $D^{\text{out}}$ contributes respectively. This gives birth to two variants of ConML: **in-MAML** which optimize $L_v$ and $D^{\text{in}}$, **out-MAML** which optimize $L_v$ and $D^{\text{out}}$. During meta-testing, we randomly sample 1000 different tasks, inside each we sample 10 different subsets each one contains $N = 10$ samples. We aggregate different subsets from the same task to form a $N = 100$ set to obtaining $\boldsymbol{e}_\tau^*$ for each task. The distribution of $D^{\text{in}}$ and $D^{\text{out}}$ are shown in Figure 2(b) and 2(e) respectively, where the dashed lines are mean values. We can find that: the alignment and discrimination ability corresponds to optimizing $D^{\text{in}}$ and $D^{\text{out}}$ respectively; the alignment and discrimination capabilities are generalizable; ConML shows the couple of both capabilities. Figure 2(c) shows the testing performance given different numbers of examples per task (shot), while the meta-leaner is trained with fixed $N = 10$. We can find that the improvement brought by $D^{\text{in}}$ is much more significant than $D^{\text{out}}$ under few-shot scenario, which indicates that alignment is closely related to the fast-adaptation ability of the meta-learner.

Table 3: Meta-testing accuracy on *mini*ImageNet.

| Category | Algorithm | Setting (5-way) | w/o ConML | ConML- | Relative Gain | Relative Time |
|---|---|---|---|---|---|---|
| Optimization-Based | MAML | 1-shot | $48.75 \pm 1.25$ | $\mathbf{56.25} \pm 0.94$ | 9.16% | 1.1× |
| | | 5-shot | $64.50 \pm 1.02$ | $\mathbf{67.37} \pm 0.97$ | | |
| | FOMAML | 1-shot | $48.12 \pm 1.40$ | $\mathbf{57.64} \pm 1.29$ | 12.65% | 1.2× |
| | | 5-shot | $63.86 \pm 0.95$ | $\mathbf{68.50} \pm 0.78$ | | |
| | Reptile | 1-shot | $49.21 \pm 0.60$ | $\mathbf{52.82} \pm 1.06$ | 5.58% | 1.5× |
| | | 5-shot | $64.31 \pm 0.97$ | $\mathbf{67.04} \pm 0.81$ | | |
| Metric-Based | MatchNet | 1-shot | $43.92 \pm 1.03$ | $\mathbf{48.75} \pm 0.88$ | 10.59% | 1.2× |
| | | 5-shot | $56.26 \pm 0.90$ | $\mathbf{62.04} \pm 0.89$ | | |
| | ProtoNet | 1-shot | $48.90 \pm 0.84$ | $\mathbf{51.03} \pm 0.91$ | 3.31% | 1.2× |
| | | 5-shot | $65.69 \pm 0.96$ | $\mathbf{67.35} \pm 0.72$ | | |
| Amortization-Based | SCNAPs | 1-shot | $53.14 \pm 0.88$ | $\mathbf{55.73} \pm 0.86$ | 3.12% | 1.3× |
| | | 5-shot | $70.43 \pm 0.76$ | $\mathbf{71.70} \pm 0.71$ | | |

Figure 2(f) shows the out-of-distribution testing performance. While meta-trained on tasks with amplitudes that uniformly distribute on $[0.1, 5]$, meta-testing is performed on tasks with amplitudes that uniformly distribute on $[0.1 + \delta, 5 + \delta]$ (the distribution shift $\delta$ is indicated as $x$-axis). We can find that the improvement brought by $D^{\text{out}}$ is notably more significant as the distribution gap grows than $D^{\text{in}}$. This indicates that discrimination is closely related to the task-level generalization ability of meta-learner. ConML takes both advantages brought by $D^{\text{in}}$ and $D^{\text{out}}$.

### 5.2 Few-Shot Image Classification

To evaluate ConML on conventional meta-learning approaches, we follow existing works [46, 17, 39, 28, 6] to evaluate the meta-learning performance with few-shot image classification problem. We consider representative meta-learning algorithms from different categories, including optimization-based: **MAML** [17], **FOMAML** [17], **Reptile** [28]; metric-based: **MatchNet** [46], **ProtoNet** [39]; and amortization-based: **SCNAPs** (Simple CNAPS) [6]. We evaluate their original meta-learning performance (**w/o ConML**) and performance meta-trained with the proposed ConML (**ConML-**). The implementation of ConML- follows the general Algorithm 2 and the specification for corresponding category in Section 3.3.

**Datasets and Settings.** We consider two few-shot image classification benchmarks: *mini*ImageNet [46] and *tiered*ImageNet [32]. 5-way 1-shot and 5-way 5-shot tasks are trained and evaluated respectively. Note that we focus on the improvement comparing ConML- and the corresponding algorithm without ConML, rather than performance comparison across different algorithms. So we conduct the experiment on each algorithm following the originally reported settings. All baselines share the same settings of hyperparameters related to the measurement of ConML: task batch size $B = 32$, inner-task sampling $K = 1$ and $\pi_\kappa(\mathcal{D}^{\text{tr}}_\tau \cup \mathcal{D}^{\text{val}}_\tau) = \mathcal{D}^{\text{tr}}_\tau$, $\phi(a, b) = 1 - {a \cdot b}/{\|a\| \|b\|}$ (cosine distance) and $\lambda = 0.1$. For other settings of hyperparameters about model architecture and training procedure, each baseline is consistent with its originally reported. Note that $K = 1$ and $\pi_\kappa(\mathcal{D}^{\text{tr}}_\tau \cup \mathcal{D}^{\text{val}}_\tau) = \mathcal{D}^{\text{tr}}_\tau$ is the most simple and efficient implementation, provided as *Efficient*-ConML in Appendix A. In this case, considering the consumption of feeding-forward neural networks in each task, Algorithm 1 takes $h = g(\mathcal{D}^{\text{tr}}_\tau; \theta)$ and $\mathcal{L}(\mathcal{D}^{\text{val}}_\tau; h)$, while ConML only introduces an additional $g(\mathcal{D}^{\text{tr}}_\tau \cup \mathcal{D}^{\text{val}}_\tau; \theta)$, which results in very comparable time consumption.

**Results.** Table 3 and 4 show the results on *mini*ImageNet and *tiered*ImageNet respectively. The relative gain is calculated in terms of the summation of 1-shot and 5-shot accuracy. The relative time is comparing the total time consumption of meta-training. Significant relative gain and very comparable relative time consumption show that ConML brings universal improvement on different meta-learning algorithms with cheap implementation.

### 5.3 In-Context Learning General Functions

Following [18], we investigate ConML on ICL by learning to learn synthetic functions including linear regression (LR), sparse linear regression (SLR), decision tree (DT) and 2-layer neural network with ReLU activation (NN). We train the GPT-2 [30]-like transformer for each function with ICL and ConML-ICL respectively and compare the inference (meta-testing) performance. We follow the same model structure, data generation and training settings [18]. We implement ConML-ICL with $K = 1$ and $\pi_\kappa([x_1, y_1, \cdots, x_n, y_n]) = [x_1, y_1, \cdots, x_{\lfloor \frac{n}{2} \rfloor}, y_{\lfloor \frac{n}{2} \rfloor}]$. To obtain the implicit representation (7), we sample $u$ from a standard normal distribution (the same with $x$'s distribution) independently in

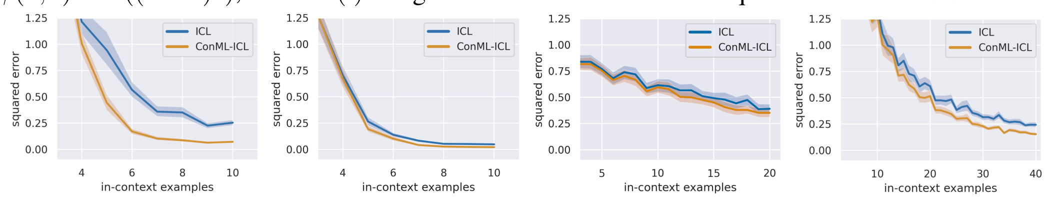
Table 4: Meta-testing accuracy on *tiered*ImageNet.

| Category | Algorithm | Setting (5-way) | w/o ConML | ConML- | Relative Gain | Relative Time |
|---|---|---|---|---|---|---|
| Optimization-Based | MAML | 1-shot | $51.39 \pm 1.31$ | $\mathbf{58.75} \pm 1.45$ | 10.07% | 1.1× |
| | | 5-shot | $68.25 \pm 0.98$ | $\mathbf{72.94} \pm 0.98$ | | |
| | FOMAML | 1-shot | $51.44 \pm 1.51$ | $\mathbf{58.21} \pm 1.22$ | 9.78% | 1.2× |
| | | 5-shot | $68.32 \pm 0.95$ | $\mathbf{73.26} \pm 0.78$ | | |
| | Reptile | 1-shot | $47.88 \pm 1.62$ | $\mathbf{55.01} \pm 1.28$ | 10.78% | 1.5× |
| | | 5-shot | $65.10 \pm 1.13$ | $\mathbf{70.15} \pm 1.00$ | | |
| Metric-Based | MatchNet | 1-shot | $48.74 \pm 1.06$ | $\mathbf{53.29} \pm 1.05$ | 11.00% | 1.2× |
| | | 5-shot | $61.30 \pm 0.94$ | $\mathbf{67.86} \pm 0.77$ | | |
| | ProtoNet | 1-shot | $52.50 \pm 0.96$ | $\mathbf{54.62} \pm 0.79$ | 3.94% | 1.2× |
| | | 5-shot | $71.03 \pm 0.74$ | $\mathbf{73.78} \pm 0.75$ | | |
| Amortization-Based | SCNAPs | 1-shot | $62.88 \pm 1.04$ | $\mathbf{65.06} \pm 0.95$ | 2.91% | 1.3× |
| | | 5-shot | $79.82 \pm 0.87$ | $\mathbf{81.79} \pm 0.80$ | | |

Table 5: Performance comparison of ConML-ICL and ICL.

| Function (max prompt len.) | LR (10 shot) | SLR (10 shot) | DT (20 shot) | NN (40 shot) |
|---|---|---|---|---|
| Rel. Min. Error | $0.42 \pm 0.09$ | $0.49 \pm .06$ | $0.81 \pm 0.12$ | $0.74 \pm 0.19$ |
| Shot Spare | $-4.68 \pm 0.45$ | $-3.94 \pm 0.62$ | $-4.22 \pm 1.29$ | $-11.25 \pm 2.07$ |

each episode. Since the output of (7) is a scalar, i.e., representation $e \in \mathbb{R}$, we adopt distance measure $\phi(a, b) = \sigma((a-b)^2)$, where $\sigma(\cdot)$ is sigmoid function to bound the squared error. $\lambda = 0.02$.




(a) LR. (b) SLR. (c) DT. (d) NN.

Figure 3: In-context learning performance.

**Results.** Figure 3 shows that varying the number of in-context examples during inference, ConML-ICL always makes more accurate predictions than ICL. Table 5 collects the two values to show the effect ConML brings to ICL: *Rel. Min. Error* is ConML-ICL's minimal inference error given different number of examples, divided by ICL's; *Shot Spare* is when ConML-ICL obtain an error no larger than ICL's minimal error, the difference between the corresponding example numbers. Note that the learning of different functions (different meta-datasets) share the same settings about ConML, which shows ConML can bring ICL universal improvement with cheap implementation. We notice that during training of LR and SLR $\lfloor \frac{n}{2} \rfloor = 5$, which happens to equals to the dimension of the regression task. This means sampling by $\pi_\kappa$ would results in the minimal sufficient information to learn the task. In this case, minimizing $D^{\text{in}}$ is particularly beneficial for the fast-adaptation ability, shown as Figure 3(a) and 3(b). This indicates that introducing prior knowledge to design the hyperparameter settings of ConML could bring more advantage. The effect of ConML for ICL is without loss of generalizability to real-world applications like pretraining large language models.

## 6 Conclusion

In this work, we propose ConML that introduce an additional supervision for episodic meta-training by exploiting task identity. The contrastive meta-objective is designed to emulate the alignment and discrimination embodied in human's fast learning ability, and measured by performing contrastive learning in the model space. Specifically, we design ConML to be integrated with the conventional episodic meta-training, and then give specifications on a wide range of meta-learning algorithms. We also reformulate training ICL into episodic meta-training to design ConML-ICL following the same principle. Empirical results show that ConML can universally and significantly improve meta-learning performance by benefiting the meta-learner's fast-adaptation and task-level generalization ability. This work lays the groundwork for contrastive meta-learning, by identifying the importance of alignment and discrimination ability of meta-learner, and practicing contrastive learning in model space. There also exists certain limitations, such as lack of investigating advanced contrastive strategy, batch- and subset- sampling strategies. We would consider these as future directions.

# References


[1] Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[2] Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

[3] Han Altae-Tran, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. Low data drug discovery with one-shot learning. *ACS Central Science*, 3(4):283–293, 2017.

[4] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

[5] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019.

[6] Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14493–14502, 2020.

[7] John Bronskill, Daniela Massiceti, Massimiliano Patacchiola, Katja Hofmann, Sebastian Nowozin, and Richard Turner. Memory efficient meta-learning with large images. *Advances in neural information processing systems*, 34:24327–24339, 2021.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[9] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[11] Wenlin Chen, Austin Tripp, and José Miguel Hernández-Lobato. Meta-learning adaptive deep kernel gaussian processes for molecular property prediction. In *International Conference on Learning Representations*, 2022.

[12] Zhe Chen. Object-based attention: A tutorial review. *Attention, Perception, & Psychophysics*, 74:784–802, 2012.

[13] Stella Christie. Learning sameness: object and relational similarity across species. *Current Opinion in Behavioral Sciences*, 37:41–46, 2021.

[14] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

[15] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227, 1979.

[16] Thomas Elsken, Benedikt Staffler, Jan Hendrik Metzen, and Frank Hutter. Meta-learning of neural architectures for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12365–12375, 2020.

[17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[18] Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598, 2022.

[19] Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International conference on machine learning*, pages 1704–1713. PMLR, 2018.

[20] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. Few-shot graph learning for molecular property prediction. In *The Web Conference*, pages 2559–2567, 2021.

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[22] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[23] John E Hummel. Object recognition. *Oxford handbook of cognitive psychology*, 810:32–46, 2013.

[24] Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. *arXiv preprint arXiv:2212.04458*, 2022.

[25] Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*, 2021.

[26] Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[27] VS Napper. Alignment of learning, teaching, and assessment. *Encyclopedia of the sciences of learning. Boston: Springer US*, pages 200–2, 2012.

[28] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[29] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[30] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[31] KI Ramachandran, Gopakumar Deepa, and Krishnan Namboori. *Computational chemistry and molecular modeling: principles and applications*. Springer Science & Business Media, 2008.

[32] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

[33] James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. *Advances in Neural Information Processing Systems*, 32, 2019.

[34] Donald Robbins. Stimulus selection in human discrimination learning and transfer. *Journal of Experimental Psychology*, 84(2):282, 1970.

[35] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[36] Johannes Schimunek, Philipp Seidl, Lukas Friedrich, Daniel Kuhn, Friedrich Rippmann, Sepp Hochreiter, and Günter Klambauer. Context-enriched molecule representations improve few-shot drug discovery. *arXiv preprint arXiv:2305.09481*, 2023.

[37] Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

[38] Albert Shaw, Wei Wei, Weiyang Liu, Le Song, and Bo Dai. Meta architecture search. *Advances in Neural Information Processing Systems*, 32, 2019.

[39] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[40] Megan Stanley, John F Bronskill, Krzysztof Maziarz, Hubert Misztela, Jessica Lanini, Marwin Segler, Nadine Schneider, and Marc Brockschmidt. FS-Mol: A few-shot learning dataset of molecules. In *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[41] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

[42] Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer, 1998.

[43] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[44] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning? *Advances in neural information processing systems*, 33:6827–6839, 2020.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[47] Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

[48] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pages 9929–9939. PMLR, 2020.

[49] Yaqing Wang, Abulikemu Abuduweili, Quanming Yao, and Dejing Dou. Property-aware relation networks for few-shot molecular property prediction. In *Advances in Neural Information Processing Systems*, pages 17441–17454, 2021.

[50] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[51] Yu-Xiong Wang and Martial Hebert. Learning to learn: Model regression networks for easy small sample learning. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pages 616–634. Springer, 2016.

[52] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Learning to model the tail. *Advances in neural information processing systems*, 30, 2017.

[53] Michael J Waring, John Arrowsmith, Andrew R Leach, Paul D Leeson, Sam Mandrell, Robert M Owen, Garry Pairaudeau, William D Pennie, Stephen D Pickett, Jibo Wang, et al. An analysis of the attrition of drug candidates from four major pharmaceutical companies. *Nature Reviews Drug discovery*, 14(7):475–486, 2015.

[54] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[55] Han-Jia Ye, Lu Ming, De-Chuan Zhan, and Wei-Lun Chao. Few-shot learning with a strong teacher. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[56] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

# A Specifications of ConML

**Algorithm 3** ConML

**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K$ and sampling strategy $\pi_\kappa$.
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = \psi(g(\kappa_k; \theta))$;
    **end for**
    Get model representation $\boldsymbol{e}_\tau^* = \psi(g(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}; \theta))$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task-specific model $h_\tau = g(\mathcal{D}_\tau^{\text{tr}}; \theta)$;
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_\tau)$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B} \sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

**Algorithm 4** *Efficient* ConML

**Input:** Task distribution $p(\tau)$, batch size $B$ (inner-task sample times $K = 1$ and sampling strategy $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}) = \mathcal{D}_\tau^{\text{tr}}$).
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    Get task-specific model $h_\tau = g(\mathcal{D}_\tau^{\text{tr}}; \theta)$, and model representation $\boldsymbol{e}_\tau^{\kappa_k} = \psi(g(\kappa_k; \theta))$;
    Get model representation $\boldsymbol{e}_\tau^* = \psi(g(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}; \theta))$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_\tau)$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B} \sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

**Algorithm 5** In-Context Learning with Contrastive Meta-Object (ConML-ICL)

**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K$ and sampling strategy $\pi_\kappa$, dummy input $u$ (probe).
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau)$;
      Get $\boldsymbol{e}_\tau^{\kappa_k} = g([\vec{\kappa_k}, u]; \theta)$;
    **end for**
    Get $\boldsymbol{e}_\tau^* = g([\vec{\mathcal{D}}_\tau, u]; \theta)$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task loss $\frac{1}{m}\sum_{i=0}^{m-1} \ell(y_{\tau,i+1}, g([\vec{\mathcal{D}}_{\tau,0:i}, x_{\tau,i+1}]; \theta))$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} \frac{1}{m}\sum_{i=0}^{m-1} \ell(y_{\tau,i+1}, g([\vec{\mathcal{D}}_{\tau,0:i}, x_{\tau,i+1}]; \theta)) + \lambda(D^{\text{in}} - D^{\text{out}})$;
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

**Algorithm 6** ConML-MAML

**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K = 1$ and sampling strategy $\pi_\kappa$
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = \theta - \nabla_\theta \mathcal{L}(\kappa_k; h_\theta)$;
    **end for**
    Get model representation $\boldsymbol{e}_\tau^* = \theta - \nabla_\theta \mathcal{L}(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}; h_\theta)$.
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task-specific model $h_{\theta - \nabla_\theta \mathcal{L}(\mathcal{D}_\tau^{\text{tr}}; \theta)}$;
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_{\theta - \nabla_\theta \mathcal{L}(\mathcal{D}_\tau^{\text{tr}}; h_\theta)})$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

**Algorithm 7** ConML-Reptile

**Input:** Task distribution $p(\tau)$, batch size $B$. (inner-task sample times $K = 1$ and sampling strategy $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}) = \mathcal{D}_\tau^{\text{tr}}$)
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau)$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = \theta - \nabla_\theta \mathcal{L}(\kappa_k; h_\theta)$;
    **end for**
    Get model representation $\boldsymbol{e}_\tau^* = \theta - \nabla_\theta \mathcal{L}(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}; h_\theta)$.
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta + \frac{1}{B}\sum_{\tau\in\boldsymbol{b}}(\boldsymbol{e}_\tau^* - \theta) - \lambda\nabla_\theta(D^{\text{in}} - D^{\text{out}})$.
**end while**

**Algorithm 8** ConML on SCNAPs

**Note:** Here $h_w$ corresponds to the feature extractor $f_\theta$; $H_\theta$ corresponds to the task encoder $g_\phi$ in [6].
**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K$ and sampling strategy $\pi_\kappa$.
Pretrain $h_w$ with the mixture of all meta-training data;
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = H_\theta(\kappa_k)$;
    **end for**
    Get model representation $\boldsymbol{e}_\tau^* = H_\theta(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task-specific model by FiLM $h_\tau = h_{w, H_\theta(\mathcal{D}_\tau^{\text{tr}})}$;
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_\tau)$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

**Algorithm 9** ConML-ProtoNet ($N$-way classification)

**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K = 1$ and sampling strategy $\pi_\kappa$
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
      Calculate prototypes $\boldsymbol{c}_j = \frac{1}{|\kappa_{k,j}|}\sum_{(x_i, y_i) \in \kappa_{k,j}} f_\theta(x_i)$ for $j = 1, \cdots, N$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = [\boldsymbol{c}_1 | \boldsymbol{c}_2 | \cdots | \boldsymbol{c}_N]$;
    **end for**
    Calculate prototypes $\boldsymbol{c}_j = \frac{1}{|\mathcal{D}_j|}\sum_{(x_i, y_i) \in \mathcal{D}_j} f_\theta(x_i)$ for $j = 1, \cdots, N$;
    Get model representation $\boldsymbol{e}_\tau^* = [\boldsymbol{c}_1 | \boldsymbol{c}_2 | \cdots | \boldsymbol{c}_N]$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task-specific model $h_{[\boldsymbol{c}_1 | \boldsymbol{c}_2 | \cdots | \boldsymbol{c}_N]}$, which gives prediction by $p(y = j \mid x) = \frac{exp(-d(f_\theta(x), \boldsymbol{c}_j))}{\sum_{j'} exp(-d(f_\theta(x), \boldsymbol{c}_{j'}))}$;
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_{[\boldsymbol{c}_1 | \boldsymbol{c}_2 | \cdots | \boldsymbol{c}_N]})$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

# B Few-shot Molecular Property Prediction

Few-shot molecular property prediction (FSMPP) is an important real-world application where meta-learning has been widely applied recently [3, 20, 49, 11, 36]. Molecular property prediction, which predicts whether desired properties will be active on given molecules, plays a crucial role in many applications like computational chemistry [31] and drug discovery [53]. As wet-lab experiments to evaluate the actual properties of molecules are expensive and risky, usually only a few labeled molecules are available for a specific property. Molecular property prediction can be naturally modeled as a few-shot learning problem [3], and meta-learning approaches has been successfully adopted for FSMPP [3, 20, 49, 11].

**Dataset and Settings.** We use FS-Mol [40], a widely studied FSMPP benchmark consisting of a large number of diverse tasks. We adopt the public data split [40]. Each training set contains 64 labeled molecules, and can be imbalanced where the number of labeled molecules from active and inactive is not equal. All remaining molecules in the task form the validation set. The performance is evaluated by $\Delta$AUPRC (change in area under the precision-recall curve) w.r.t. a random classifier [40], averaged across meta-testing tasks.

**Baselines.** We consider the following meta-learning-based FSMPP approaches: **MAML**, **ProtoNet**, **CNP**, **IterRefLSTM**, **PAR**, **ADKF-IFT**. Note that MHNfs [36] is not included as it uses additional reference molecules from external datasets, which leads to unfair comparison, and ADKF-IFT is the SOTA approach in literature. All baselines share the same GNN-based encoder provided by the benchmark to meta-train from scratch, which maps molecular graphs to embedding vectors.

**Algorithm 10** Hypro

**Note:** The main-network consists of two modules [40]: the molecular encoder $f_\theta$ and the prototypical network classifier $h_\theta$.
**Input:** Task distribution $p(\tau)$, batch size $B$.
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    Encode all molecules $f_\theta(x)$ for $x \in \mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}}$
    Get task-specific parameters $\alpha_\tau = H_\theta(\{(f_\theta(x_i), y_i)\}_{(x_i,y_i)\in\mathcal{D}_\tau^{\text{tr}}})$;
    Modulate all molecular embedding with $\alpha_\tau$ by FiLM, and classify with $h_\theta$; (denote the function of this step as $h_{\theta,\alpha_\tau}$)
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_{\theta,\alpha_\tau})$;
  **end for**
  $L_v = \frac{1}{B}\sum_{\tau\in\boldsymbol{b}} \mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_{\theta,\alpha_\tau})$
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L_v$.
**end while**

We introduce a new baseline **ConML-Hypro**, which achieves SOTA performance by incorporating ConML with a simple backbone, **Hypro**. It is an amortization-based model built by modifying the ProtoNet backbone, by plugging-in a hypernetwork $H$ with a set-encoder structure, i.e., $H(\mathcal{D}) = \texttt{MLP}_2(\frac{1}{|\mathcal{D}|}\sum_{\mathcal{D}} \texttt{MLP}_1([x_i \mid y_i]))$. We input the embedding vectors in $\mathcal{D}^{\text{tr}}$ to the hypernetwork, and take the output to modulate embedding vectors through FiLM before classification. This hypernetwork and modulation is typical in amortization-based models. Viewing Hypro as an amortization-based model, we apply the specification of ConML to form ConML-Hypro. The detailed procedure to train Hypro and ConML-Hypro are provided in Algorithm 10 and 11. The structure of $H$ is provided in Table 6, and two such hypernetworks are used for generate parameters for FiLM function. We implement ConML with $B = 16$, $\phi(a, b) = 1 - a\cdot b/\|a\|\|b\|$ (cosine distance) and $\lambda = 0.1$. As for the sampling strategy $\pi_\kappa$ and times $K$, for every task, we sample subset with different sizes, including each $m \in \{4, 8, 16, 32, 64\}$, for $128/m$ times respectively. A $m$-sized subset contains $m/2$ positive and $m/2$ negative samples sampled randomly. The other hyperparameters of model structure and training procedure follow the benchmark's default setting [40].

**Results.** Table 7 shows the results. ConML-Hypro shows advantage over SOTA approach under all meta-testing scenarios with different shots. Comparing Hypro and ProtoNet, we can find the

**Algorithm 11** ConML-Hypro

**Note:** Refer to Algorithm 10 for details about $H_\theta(\mathcal{D})$ and $h_{\theta,\alpha}$.
**Input:** Task distribution $p(\tau)$, batch size $B$, inner-task sample times $K$ and sampling strategy $\pi_\kappa$.
**while** Not converged **do**
  Sample a batch of tasks $\boldsymbol{b} \sim p^B(\tau)$.
  **for** All $\tau \in \boldsymbol{b}$ **do**
    **for** $k = 1, 2, \cdots, K$ **do**
      Sample $\kappa_k$ from $\pi_\kappa(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
      Get model representation $\boldsymbol{e}_\tau^{\kappa_k} = H_\theta(\kappa_k)$;
    **end for**
    Get model representation $\boldsymbol{e}_\tau^{*} = H_\theta(\mathcal{D}_\tau^{\text{tr}} \cup \mathcal{D}_\tau^{\text{val}})$;
    Get inner-task distance $D_\tau^{\text{in}}$ by (2);
    Get task-specific model $h_{\theta, H_\theta(\mathcal{D}_\tau^{\text{tr}})}$;
    Get validation loss $\mathcal{L}(\mathcal{D}_\tau^{\text{val}}; h_{\theta, H_\theta(\mathcal{D}_\tau^{\text{tr}})})$;
  **end for**
  Get $D^{\text{in}} = \frac{1}{B}\sum_{\tau \in \boldsymbol{b}} D_\tau^{\text{in}}$ and $D^{\text{out}}$ by (3);
  Get loss $L$ by (4);
  Update $\theta$ by $\theta \leftarrow \theta - \nabla_\theta L$.
**end while**

Table 6: Hypernetwork structure in Hypro and ConML-Hypro

| | Layers | Output dimension |
|---|---|---|
| $\texttt{MLP}_1$ | input $[x_i \mid y_i]$ (dim=2562), fully connected, LeakyReLU | 2560 |
| | $2\times$ fully connected with with residual skip connection | 2560 |
| $\texttt{MLP}_2$ | $2\times$fully connected with residual skip connection, LeakyReLU | 2560 |

introduced hypernetwork can brings notable improvement. Comparing ConML-Hypro and Hypro, we can find the effect of ConML is significant.

Table 7: Few-shot molecular property prediction performance ($\Delta$AUPRC) on FS-Mol. † indicates result from [36]. $*$ indicates new approach proposed in this paper.

| | 2-shot | 4-shot | 8-shot | 16-shot |
|---|---|---|---|---|
| MAML | $.009 \pm .006$ | $.125 \pm .009$ | $.146 \pm .007$ | $.159 \pm .009$ |
| PAR | $.124 \pm .007$ | $.140 \pm .005$ | $.149 \pm .009$ | $.164 \pm .008$ |
| ProtoNet | $.117 \pm .006$ | $.142 \pm .007$ | $.175 \pm .006$ | $.206 \pm .008$ |
| CNP | $.139 \pm .004$ | $.155 \pm .008$ | $.174 \pm .006$ | $.187 \pm .009$ |
| Hypro$^*$ | $.122 \pm .007$ | $.150 \pm .006$ | $.185 \pm .008$ | $.216 \pm .007$ |
| IterRefLSTM$^\dagger$ | - | - | - | $.234 \pm .010$ |
| ADKF-IFT | $.131 \pm .007$ | $.166 \pm .005$ | $.202 \pm .006$ | $.234 \pm .009$ |
| ConML-Hypro$^*$ | $\mathbf{.175} \pm .006$ | $\mathbf{.196} \pm .006$ | $\mathbf{.218} \pm .005$ | $\mathbf{.239} \pm .007$ |

# NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: **[TODO]**

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer:[Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer:[Yes]

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to

generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.