Clustered Meta-Learning

Behzad Haghgoo BHAGHGOO@STANFORD.EDU
Megumi Sano MEGSANO@STANFORD.EDU
Advay Pal ADVAYPAL@STANFORD.EDU

Abstract

Model-Agnostic Meta-Learning (MAML) has shown state-of-the-art results in a wide variety of meta-learning tasks. However one may expect the inner adaptation performance of MAML to degrade on task distributions with a high diversity of tasks. We hypothesise that using a single meta-parameter as the initialization for all tasks becomes a bottleneck for MAML in such situations and propose Clustered Meta-Learning (CML), which uses an adaptation-based distance function to cluster tasks and leverage multiple meta-parameters, one for each cluster. The key underlying theory is that in having multiple meta parameters, each meta-parameter does not have to learn to generalize across tasks that may be very different (far-apart in the parameter space). At the same time, by maintaining multiple such meta-parameters, the algorithm as a whole is able to maintain the ability to generalise to new tasks, as when seeing a new task, one only needs to identify which group(s) it is a part of, and use the meta-parameter for that group, or if the task does not belong to any group, create a new group on its own. We show that our algorithm outperforms MAML in settings with multimodal task distributions (where the tasks can be clustered into groups of highly similar tasks).

We structure our paper as follows: First, we empirically show that given access to ideal clustering, having a meta-parameter per cluster helps increase the performance of MAML in multimodal task distributions with the same number of training steps. We do this by comparing the performance of MAML trained on a multimodal task distribution containing tasks from two environments with the performance of an oracle: two separate MAML meta-learners trained individually on each environment (at test time we decide which meta-learner to use based on the environment it came from). Following this, we introduce our *Adaptation Distance Function*, which aims to compute task similarity as being proportional to the number of inner update steps required between tasks and show its efficacy in clustering tasks such that MAML can achieve faster adaptation. Finally, we introduce Clustered Meta-Learning, an algorithm that uses our Adaptation Distance Function to cluster tasks. We compare CML's performance with our MAML baseline and the hand-crafted clustering based on an oracle. We find that CML out-performs MAML and even the oracle with access to ideal clustering.

1. Introduction

Model-Agnostic Meta Learning (MAML) has shown promising results in a wide variety of both meta-supervised learning and meta-reinforcement learning tasks (Finn et al., 2017a,b). However, most reinforcement learning settings where MAML is used deal with a narrow task distribution (Yu et al., 2019), making MAML incapable of generalising well to new tasks. Furthermore MAML has been shown to lose its high performance when trained on more diverse task distributions (Yu et al., 2019).

Real-world settings are often comprised of a hierarchy of tasks. Rather than tasks being distributed uniformly in the parameter space, which may often be an assumption in commonly used simulated environment settings (eg. uniform sampling of goals in the Mujoco environment), real-world tasks have an inherent *hierarchy*, in which one can identify groups of highly similar tasks e.g. *window opening and window closing*, and stacking and unstacking in MetaWorld (Yu et al., 2019). As such, we aim to develop a meta-learning algorithm that can leverage the structure of a multi-modal task distribution in order to outperform MAML in such situations.

We hypothesize that one of the main bottlenecks in the generalizability of MAML to multimodal task distributions is in the use of a singular meta-parameter. (We refer to a set of meta-parameters θ used to parameterize a model as one meta-parameter in this paper, as to avoid confusion with our method which obtains multiple models, each with one θ.). MAML seeks to find the optimal point in the meta-parameter space that is able to reach the task-specific parameters for all tasks in a small number of gradient updates.

Our proposed algorithm, Clustered Meta-Learning (CML), aims to improve the performance of MAML on multimodal task distributions by maintaining a set of meta parameters, one for each cluster of similar tasks. The key idea here is that in having multiple meta parameters, each meta-parameter does not have to learn to generalise across tasks that may be very different (far-apart in the parameter space), something which we hypothesised was the cause of MAML's performance degrading in the multimodal task distribution setting. At the same time, by maintaining multiple multiple such meta-parameters, the algorithm as a whole is able to maintain an ability to generalise to new tasks, as when seeing a new task, one only needs to identify which group(s) it is a part of, and use the meta-parameter for that group.

Similar approaches have been proposed to leverage task structure in improving the sample efficiency of meta-learning, e.g. by learning task embeddings (Vuorio et al., 2018; Hausman et al., 2018) and by constructing latent variables (Rakelly et al., 2019). Our primary contribution differs from these in that we propose a task distance function which is directly optimized for clustering tasks such that the task-specific parameters for tasks within the same cluster can be arrived at using a small number of gradient steps from a single meta-parameter. (See Algorithm 1).

Specifically, in this work, we aim to address the following main questions: (1) What is MAML's performance in scenarios with multimodal task distributions? (2) Assuming access to ideal clusters, does having cluster-wise meta-parameters help improve performance over MAML? (3) How effective is CML in finding these clusters based on our proposed distance function? In order to further understand our method, with regards to (3), we also investigate how the performance of CML changes with different initialization methods for the initial set of meta-parameters (See Section 3 for details).

2. Related Work

In this section, we provide a review of prior work that explores the problem of leveraging task distribution structure to improve performance.

Multimodal Model-Agnostic Meta-Learning via Task-Aware Modulation

Similarly inspired by the hypothesis that using multiple meta-learners instead of a single meta-learner would allow for better performance in scenarios with a multimodal task distribution, Vuorio et al. propose a two-stage algorithm which first computes the identity of a newly sampled task, and then conditions on the inferred task mode to modulate the meta-parameter such that it better fits the inferred mode (Vuorio et al., 2018). More specifically, they use two networks: (1) a modulation network which takes as input data samples and generates task embeddings that are then used to generate parameters, and (2) a task network modulated by these parameters, which then adapts to the target task.

While they explore a diverse range of tasks including regression, few-shot image classification, and reinforcement learning tasks, since our work tackles reinforcement learning in particular, here we specifically focus on their approach to reinforcement learning tasks. One of the main differences between their approach and ours is that their approach separates task inference from adaptation, by first using an initial policy to collect a batch of trajectories which are fed into the modulation network to infer task modes. Thus, this approach relies on access to the correspondence between the task identity of a newly encountered task during the meta-test phase and its batch of trajectories collected at the meta-training step, whereas our algorithm does not rely on this correspondence.

Moreover, although their two-stage algorithm outperforms MAML in the chosen RL tasks such as Point Mass, Reacher, and Ant, where the multimodal task distribution is reflected in a multimodal goal distribution, one may expect that it may not generalize as well to more complex multimodal task disributions such as Meta-World (Yu et al., 2019), since the initial policy used to infer the task modes may not be good enough, which could lead to suboptimal learning of effective task embeddings. Finally, their task mode inference is less directly optimized for quicker inner adaptation compared to ours.

Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables

Rakelly et al. introduce PEARL (Probabilistic embeddings for actor-critic RL) which represents task contexts with probabilistic latent variables to enable reasoning over task uncertainty (Rakelly et al., 2019). Specifically, given a set of training tasks, the algorithm learns a policy that adapts to a given task by conditioning on the context, learned to capture minimal sufficient statistics of task information without modeling irrelevant dependencies from a history of past transitions. The rationale is that the data used to train the context encoder does not need to be the same as the data used to learn the policy, which allows for off-policy meta-training and thus requires fewer environment interactions. Their approach shows improvement in sample efficiency, both in terms of the number of samples from previous experience as well as the number of samples required in a newly encountered task for adaptation. Hausman et al. (2018) compose tasks via embedding space, different aim (Hausman et al., 2018).

It is important to note that while the high-level goal of achieving faster inner adaptation is shared with our approach, the usefulness of PEARL is not specific to the problem of multimodal task distributions. One can imagine that context variables are a more general form of task clustering done by Vuorio et al. and by our work, since its usefulness scales with how much structure there is in the task distribution, but the structure need not necessarily be multimodal, while our work focuses specifically on the problem of multimodal task distributions.

Modular Meta-Learning

Alet et al. attempt to tackle the problem of generalizing to a diverse set of tasks by learning a set of modules that can be recombined to perform a new task (Alet et al., 2018). Instead of finding a single meta-parameter θ as a reasonable starting point for a few gradient descent steps in each new task, the rationale is that a new task's optimal parameters should be able to be expressed as some

combination of modules learned on previous tasks, exploiting the compositional nature of real-world tasks and the power of combinatorial generalization.

In our case, the high-level assumption is that given a set of tasks, there exists some hierarchical structure, wherein at each level of the hierarchy, the set of tasks can be grouped into further subsets (clusters) of tasks such that within-cluster similarity and across-cluster dissimilarity hold (defined by some distance metric, see Section 3.1.1).

3. Methods

This section provides a detailed explanation of our algorithm and the steps we took in order to verify it works. Some of these steps are still in-progress and the results section will provide more details about which steps have already been executed as well as what should be done next.

3.1 Clustered Meta Learning

This section provides a description of our final algorithm, and the section following it will describe our rationale behind each part.

Suppose we have n tasks at hand, $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n\}$, sampled from some distribution $\mathcal{P}(\mathcal{T})$. We specifically investigate a setting in which $\mathcal{P}$ is multimodal, such that there exists $m < n$ clusters of tasks with high within-cluster similarity. We maintain a set Θ (initially containing some first meta-parameter θ) that represents the meta-parameters for each group of tasks. Whenever we encounter a new task, we either create a new meta-parameter for that task and hence a new cluster, or we use an existing meta-parameter from Θ and slightly fine-tune it.

More specifically, for all $\theta \in \Theta$, we perform gradient steps to get corresponding task-specific parameters ϕs for that task. Then we find the ϕ with the highest average return over a rollout of the policy (for a specific length) and see how much it has improved in the last gradient step of the mentioned process. If it is lower than some threshold ϵ (a hyperparameter), we believe the model has converged to that task and thus we don't create a new θ. In the other case if the model is not sufficiently close to convergence, we continue training the best ϕ on this task until convergence and store it as a new θ. This way, when facing a new task at test time, there is a higher probability that we have a closer θ to that task compared to MAML (we closely discuss the meaning of "closer" in the next section). Figure 1 pictorially demonstrates the general framework for generating θs for each group of tasks. For a more detailed and precise outline of our meta-training algorithm, see Algorithm 1.

3.1.1 The Adaptation Distance Function

Our algorithm is created on the basis that understanding the hierarchy of tasks and developing multiple meta-parameters based on them can increase the performance of the MAML algorithm. In order to do so, it is important to have a way of measuring similarity between various tasks so that similar tasks can be associated to the same meta-parameter.

In our algorithm, the ideal distance function used to assign a new task to a cluster would be one that clusters the tasks in a way such that each group of tasks are easily learnable with their respective meta-parameter via MAML. More formally, given a fixed $\Theta = \{\theta_1, \ldots, \theta_k\}$, analogous to the centroids of existing clusters, and a fixed set of tasks, $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$, an ideal distance function would maximize the sum of the performance (measured using average reward over a certain policy roll-out length or the average success rate of roll-outs) of the task-specific parameter on each task after the inner gradient step(s):

$$\mathbb{R} = \sum_{i=1}^{n} \mathcal{R}_{\mathcal{T}_i}(\phi_i^*)$$

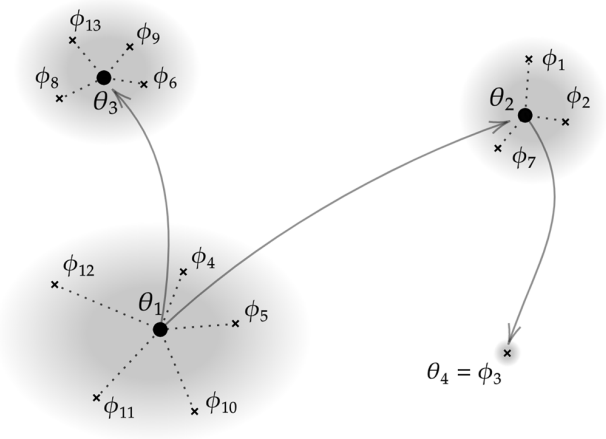

Figure 1: Diagram of our Clustered Meta-Learning (CML) algorithm. CML finds clusters of tasks where each cluster has a meta-parameter that enables it to quickly generalize to tasks in that cluster. Note that we do not need to specify the exact number of clusters in advance.

where $\mathcal{R}_{\mathcal{T}_i}$ is the average reward for task i and ϕ_i^* is the task specific parameters used for solving task i. Provided that we already have a set of meta-parameters Θ, the obvious solution for optimizing the this term is

$$\max \mathbb{R} = \sum_{i=1}^{n} \arg\max_{\theta \in \Theta} \mathcal{R}_{\mathcal{T}_i}(\phi_i(\theta))$$

where ϕ_i is a function that takes $\theta \in \Theta$ as input and returns the corresponding task parameters for task i by doing MAML inner-updates.

Thus, if we are provided with Θ, the optimal way for assigning a meta-parameter to task $\mathcal{T}_i$ is by doing the inner-loop updates on each $\theta \in \Theta$ and choosing the one that maximises $\mathcal{R}_{\mathcal{T}_i}$, and we refer to this as the Adaptation Distance Function. While we have not mathematically shown that previous methods aim to approximate this objective, we believe it is intuitively clear that our approach is a more direct way to optimize for the objective and leave theoretical derivations for future work.

3.1.2 Obtaining Meta-Parameters

As we assumed that we are given a set of meta-parameters Θ, the next question is how can we obtain such a set at the beginning of our algorithm? We explore the following strategies:

Random initialization. We randomly initialize the initial set of meta-parameters. While this method requires the least pre-training, we also expect this to perform poorly, because both meta-parameters are not optimized for any of the tasks, which may result in random cluster assignments.

Task-specific parameters. We can pre-train a set of models, each on one task sampled from the task distribution to obtain a task-specific parameter. The rationale behind this is that this is a simple way to obtain meta-parameter initializations that just requires access to $|\Theta|$ number of tasks. Assuming that the task distribution is multimodal, the obtained task-specific parameters should be

close to the mode of the cluster the task belongs to and far from other modes, which should result in better clustering.

Algorithm 1: Clustered Meta-Learning (CML)

Input: $p(T)$ **(the distribution of tasks),** α**,** β**,** ϵ**,** $\Theta = \{\theta_1, \theta_2, ..\theta_m\}$ **where** θ_i **represents our initialised parameters for each cluster while** *not done* **do**

- Sample K tasks $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_\mathcal{K}\} \sim p(T)$
- Set total gradient update for each cluster to zero: $U_{\theta_j} = 0$
- Set the counter for examples of each cluster to zero $C_{\theta_j} = 0$
- **for** $\mathcal{T}_i \in \mathcal{T}$ **do**
  - **for** $\theta_j \in \Theta$ **do**
    - Sample M trajectories $D = \{(x_1, a_1, ...x_H)\}$ using f_{θ_j}
    - Evaluate $\nabla_{\theta_j} \mathcal{L}_{T_i}(f_{\theta_j})$ using D where $\mathcal{L}_{T_i}(f_\phi) = -\mathbb{E}_{x_t, a_t \sim f_\phi, qT_i}[\sum_{t=1}^{H} R_i(x_t, a_t)]$
    - Compute adapted parameters using gradient descent ${\theta_j}' = \theta_j - \alpha \nabla_{\theta_j} \mathcal{L}_{T_i}(f_{\theta_j})$
    - Sample trajectories $D'_k = \{(x_1, a_1, ...x_H)\}$ using $f_{{\theta_j}'}$ in T_i
  - **end**
  - $\theta' = \text{argmax}_k \mathcal{R}_{T_i}(f_{\theta'_i})$ using D'_k where $\mathcal{R}_{T_i}$ represents the average reward over a fixed policy rollout length.
  - Let $D' = D'_k$ and $\theta = \theta_i$ for the k that corresponds to the $argmax$.
  - Let $\theta'' := \theta' - \beta \nabla_{\theta'} \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta'})$ using D' where $\mathcal{L}_{T_i}$ is the same as above
  - Update $U_{\theta_i} \mathrel{+}= \beta \nabla_{\theta'} \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta'})$
  - $C_{\theta_j} \mathrel{+}= 1$
- **end**
- **for** $\theta_i \sim \Theta$ **do**
  - $\theta_i := \theta_i - U_{\theta_i} / C_{\theta_j}$
- **end**

end

Algorithm 2: Flexible Clustered Meta-Learning (FCML)

Input: $p(T)$ **(the distribution of tasks),** α**,** β**,** ϵ**,** $\Theta = \{\theta\}$ **where** θ **represents our initial set of meta parameters while** *not done* **do**

- Sample K tasks $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_\mathcal{K}\} \sim p(T)$
- Set total gradient update for each cluster to zero: $U_{\theta_j} = 0$
- Set the counter for examples of each cluster to zero $C_{\theta_j} = 0$
- **for** $\mathcal{T}_i \in \mathcal{T}$ **do**
  - **for** $\theta_j \in \Theta$ **do**
    - Sample M trajectories $D = \{(x_1, a_1, ...x_H)\}$ using f_{θ_j}
    - Evaluate $\nabla_{\theta_j} \mathcal{L}_{T_i}(f_{\theta_j})$ using D where $\mathcal{L}_{T_i}(f_\phi) = -\mathbb{E}_{x_t, a_t \sim f_\phi, qT_i}[\sum_{t=1}^{H} R_i(x_t, a_t)]$
    - Compute adapted parameters using gradient descent ${\theta_j}' = \theta_j - \alpha \nabla_{\theta_j} \mathcal{L}_{T_i}(f_{\theta_j})$
    - Sample trajectories $D'_k = \{(x_1, a_1, ...x_H)\}$ using $f_{{\theta_j}'}$ in T_i
  - **end**
  - $\theta' = \text{argmax}_k \mathcal{L}_{T_i}(f_{\theta'_i})$ using D'_k where $\mathcal{R}_{T_i}$ represents the average reward over a fixed policy rollout length.
  - Let $D' = D'_k$ and $\theta = \theta_i$ for the k that corresponds to the $argmax$.
  - Let $\theta'' := \theta' - \beta \nabla_{\theta'} \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta'})$ using D' where $\mathcal{L}_{T_i}$ is the same as above
  - **if** $\frac{\mathcal{R}_{T_i}(f_\theta) - \mathcal{R}_{T_i}(f_{\theta''})}{\mathcal{R}_{T_i}(f_\theta)} \leq \epsilon$ *using* D' **then**
    - we update $U_{\theta_i} \mathrel{+}= \beta \nabla_{\theta'} \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta'})$
    - $C_{\theta_j} \mathrel{+}= 1$
  - **end**
  - **else**
    - we tune θ'' with more gradient steps till convergence and then add θ'' to Θ
  - **end**
- **end**

end

4. Experimental setup

4.1 Environment

We used the AntRandDirec and CheetahRandDirec environments from OpenAI Gym (Brockman et al., 2016). The idea throughout the rest of the paper is that our environments are different enough such that each environment corresponds to a centroid in the true clusters of tasks, and given an environment, tasks sampled from that environment belong to the cluster with that environment as the centroid.

We adapt our method for obtaining meta-parameters (see Section 3) to this task distribution by initializing two random meta-parameters for the random initialization strategy and two task-specific parameters, each one trained from a task from one of the two environments for 50 iterations, for the task-specific initialization strategy.

These environments have slightly different observation and action spaces and thus we padded them to reach the same dimensions (Cheetah has an observation space of 26 and an action space of 6, while Ant has an observation space of 111 and an action space of 8).

4.2 Architecture and hyperparameters

We use a two-layer fully connected network with 64 hidden units as our policy network and the Trust Region Policy Optimization (TRPO) algorithm for optimizing it (Schulman et al., 2015). We borrow the code for the policies and meta-learning algorithms from (Rothfuss et al., 2018). We sample 40 tasks per each update step, and for each task, we sample 20 trajectories. We use an inner learning rate of 0.1, 2 inner gradient steps, and an outer learning rate of 1e-3. In order to make the comparison with MAML algorithm more fair in terms of training data, we keep $\mathcal{K}$ (the number of tasks sampled at each iteration) the same for MAML and CML.

5. Results

In this section, we describe three sets of experiments we ran to test our method.

5.1 Baseline and oracle

First, we are interested in the performance of our baseline, MAML, and our oracle, Multi-MAML, which is given access to the ideal clustering. We test these two methods on our artificially created multimodal task distribution by training the MAML algorithm on the two environments together as our baseline and training two different MAML algorithms, each on one of the environments, as our Multi-MAML oracle. Our results show that if the clustering is done properly, it is possible to get a substantial performance gain on MAML (See `Multi-MAML (Oracle)` and `MAML (Baseline)` curves in Figure 2).

5.2 Testing the Adaptation Distance Function

After testing the efficacy of having a meta-parameter per cluster of tasks, under the assumption of oracle access to ideal clustering, the crucial question becomes: Does our Adaptation Distance Function allow us to identify such clusters (or good enough clusters)? To do this, we perform the following two experiments.

5.2.1 Clustering score on pre-trained meta-parameters

To empirically test our distance function, we train 2 MAML algorithms on two separate environments and see if using the converged meta-parameters of each environment will lead to proper clustering.

In our task distribution setting, proper clustering is defined by accurate detection of which of the two environments a task is sampled from. In order to track the quality of our clustering, we use the following metric:

$$\mathcal{C} = 2\left|\mathbb{E}\left(|I_{\text{true}} - I_{\text{pred}}|\right) - \frac{1}{2}\right|$$

Where I_{true} is the true environment index for each example and I_{pred} is the predicted index based on our distance function for each example. $\mathcal{C} = 1$ indicates a clustering that conforms with I_{true} and $\mathcal{C} = 0$ implies random clustering. This is a temporary metric for the mentioned experiments as it is only suitable for $|\Theta| = 2$. This empirical validation tells us whether the differences between task-distances from our centroids is meaningful or not. We find that we achieve a clustering score of **1.0** after switching to CML, suggesting that our distance function successfully separates tasks from each environment from each other and assigns the correct meta-parameter to the corresponding tasks.

5.2.2 Testing trained meta-parameters on an unseen environment

In addition to computing the above clustering score, we also wanted to ensure that the two environments we chose are not too "close" in our adaptation distance space i.e. the way we construct the multimodal task distribution is valid. To do so, we train the MAML algorithm on one environment, and then switch the environment at 200 iterations to the other environment, such that the algorithm must adapt to the tasks from an unseen environment with the already trained meta-parameter.

Figure 3 shows a sudden jump in average return when switching the environment from Ant to Cheetah and vice versa. This shows that the loss for a random selection of tasks in each environment is substantially different from the loss on the other environment and thus, two given theta centroids would partition a set of given tasks from the two environments accurately.

5.3 Performance comparison

Finally, we evaluate the performance of our algorithm (both with randomly initialized meta-parameters and with task-specifically initialized meta-parameters) against MAML and Multi-MAML using average return across training iterations.

First, we find that both of our methods outperform MAML. Moreover, contrary to our expectations, we find that CML with random initialization outperforms the oracle (Multi-MAML) and CML with task-specific initialization performs as well as Multi-MAML (See Figure 2). Although our results are too early for us to come to a conclusion, we believe this might because of the fact that our environment-based clustering for the oracle is not necessarily the optimal clustering for fast learning. This suggests the possibility that our method was able to find clusters that were more useful for inner adaptation than the ones we had artificially created. This is also supported by the observation that the clustering score for our algorithm (both with randomly initialized meta-parameters and with task-specifically initialized meta-parameters) is relatively low, because the clustering score is based on the artificial clusters we specify i.e. each environment belonging to one cluster. Regardless, our results suggest that our algorithm is able to identify clusters such that it can perform faster inner adaptation, which was the main goal of our approach.

6. Conclusion and Future Directions

We presented Clustered Meta-Learning (CML), a method for meta-learning with multiple meta-parameters that increases the versatility of the MAML algorithm. In addition, we introduced the

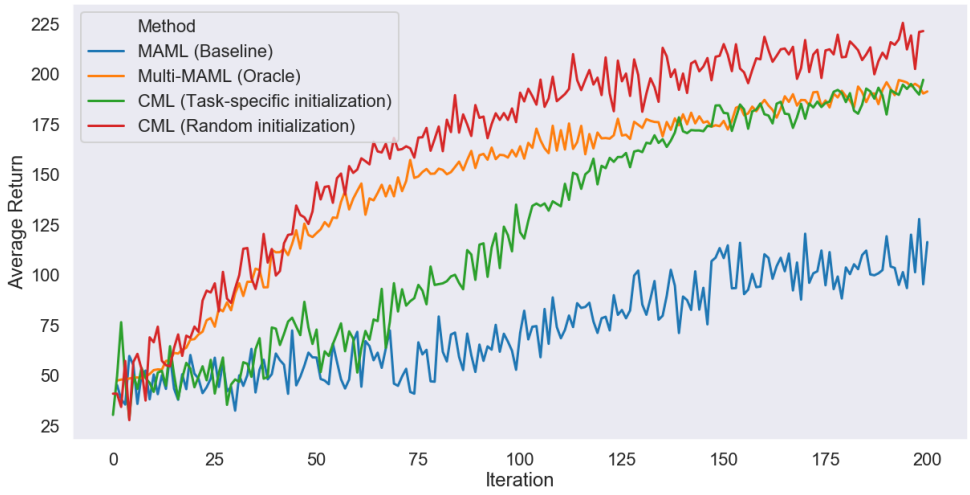

Figure 2: Average return plotted as a function of iterations for MAML (baseline), Multi-MAML (oracle), MAML (baseline), CML (Task-specific initialization), and CML (Random initialization). We find that CML outperforms both MAML and eventually the Multi-MAML oracle. CML is able to identify clusters such that the cluster-wise meta-parameters can adapt to newly encountered tasks quickly.

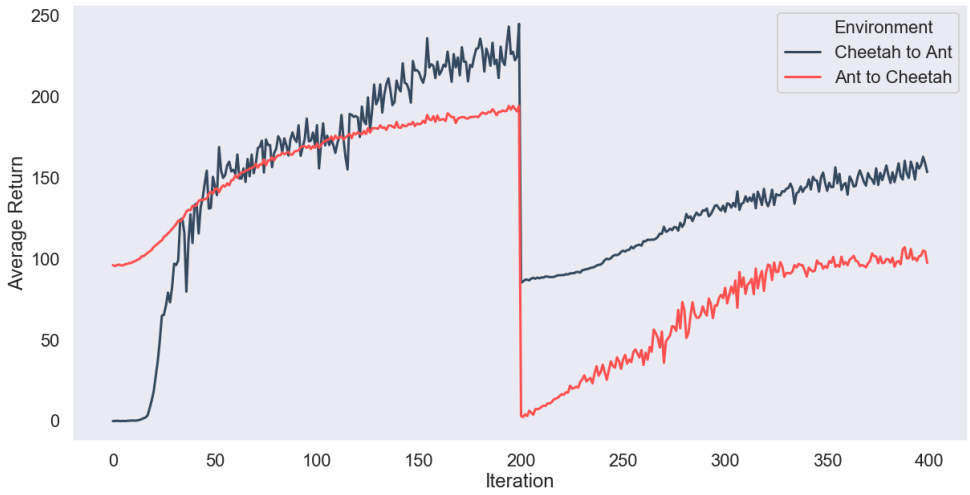

Figure 3: Average return plotted as a function of iterations while switching between environments: Two MAML algorithms are trained separately, one on the Mujoco Ant environment and one on the Mujoco Cheetah environment. At iteration 200, the environments are switched and the changes in average reward is monitored. It is evident that a MAML algorithm is significantly better on its training environment compared to another environment showing that our adaptation distance function is a practical way for assigning tasks to algorithms in CML algorithm.

Adaptation Distance Function, which measures similarity of different tasks based on their respective loss after inner-updates and showed its efficacy for CML. Our experiments show that CML outperforms MAML and multi-MAML settings and show how the Adaptation Distance Function helps with clustering.

Our results are still early and there are many avenues for further investigation. In particular, the following areas are key to understanding this problem space further:

- Implementing CML with flexible number of clusters and experimenting with various thresholding methods for deciding whether to add a newly encountered task to an existing cluster or create a new cluster. Having a fixed number of clusters can be a short-coming especially in computationally expensive networks and thus an approach similar to Algorithm 2 can significantly increase the applicability of CML.

- Further work on understanding the nature of the Adaptation Distance Function and possibly coming up with a valid *metric* for defining how different tasks are related. (Currently, the Adaptation Distance Function is not a valid distance metric.)

- Studying the limitations of MAML and CML under constraints on the total number of allowed parameters. The multitude of architectures that are possible for a given number of parameters, makes such comparison challenging but it can lead to insights about when clustering is helpful and when it is not. Note that while we did not control for the total number of parameters between MAML and CML, the total number of parameters for Multi-MAML (oracle) and CML are the same. CML outperforming Multi-MAML suggests that assuming that clustering is helpful, our algorithm is able to do it effectively i.e. in a way that leads to faster inner adaptation.

- Implementing CML on Meta-World and comparing it with state-of-the-art benchmarks presented by Yu et al. (2019). We believe that Meta-World is a particularly good environment for testing our algorithm due to its multimodal nature.

Acknowledgments

We would like to thank Suraj Nair and Tianhe Yu for helpful discussion.

All code and materials available at:
`https://github.com/behzadhaghgoo/cml`

Appendix

Notes on technical difficulties of the project

- For most of the quarter we were trying to use the MetaWorld environment (Yu et al., 2019) but due to technical difficulties of setting up, we ended up switching to OpenAI Gym.

- Up until after the poster session, we were also using the surrogate loss in TRPO instead of average return for the clustering distance function and getting results that did not match our theory. We realized late in the quarter that the loss is not as correlated with average return as we thought it was (See Figure 4) and changed our distance function to use average return, which quickly achieved good results.

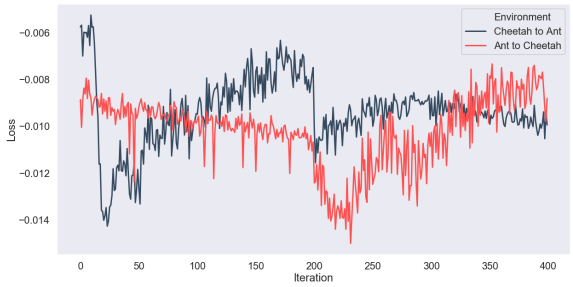

Figure 4: Exactly the same plot as Figure 3, except instead of average return, we plot the surrogate loss of TRPO. We find that while we see the sharp change that we saw in average return, the rest of the pattern in the loss is not correlated with reward i.e. while we expect the loss to decrease as the reward increases, sometimes the loss increases during iterations in which reward increases.

References

Ferran Alet, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. Modular meta-learning. *CoRR*, abs/1806.10166, 2018. URL `http://arxiv.org/abs/1806.10166`.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017a.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. *arXiv preprint arXiv:1709.04905*, 2017b.

Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller. Learning an embedding space for transferable robot skills. 2018.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *arXiv preprint arXiv:1903.08254*, 2019.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. Promp: Proximal meta-policy search. *arXiv preprint arXiv:1810.06784*, 2018.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

Risto Vuorio, Shao-Hua Sun, Hexiang Hu, and Joseph J. Lim. Toward multimodal model-agnostic meta-learning. *CoRR*, abs/1812.07172, 2018. URL `http://arxiv.org/abs/1812.07172`.

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. *arXiv preprint arXiv:1910.10897*, 2019.