A REPORT ON OFFLINE META-REINFORCEMENT LEARNING WITH ADVANTAGE WEIGHTING[1]

Jeevana Kruthi ***Aparna Sakshi*** ***Sanjana Kasarla***

Indian Institute of Technology, Kharagpur

1. INTRODUCTION

Meta-learning is a branch of machine learning which aims to quickly adapt models, to perform new tasks by learning an underlying structure across related tasks. Meta-reinforcement learning algorithms leverage experience across many tasks to learn fast and effective reinforcement learning strategies. In general the meta reinforcement learning algorithms are assumed to be able to collect the data online during the meta-training phase. Developing an offline meta-RL method would allow to leverage existing data from any source, making them easier to scale to real-world problems where large amounts of data might be necessary to generalize broadly. Like supervised learning strategy of pre-training a model on a large batch of fixed, pre-collected data (possibly from various tasks) and fine-tuning the model to a new task with relatively little data, in offline meta-RL, we meta-train on fixed, pre-collected data from several tasks in order to adapt to a new task with a very small amount of data from the new task.

In this paper an offline meta-RL problem setting is proposed and a corresponding algorithm that uses only offline (or batch) experience from a set of training tasks to enable efficient transfer to new tasks without requiring any further interaction with either the training or testing environments.

The offline setting does not allow additional data collection during training, it highlights the desirability of a consistent meta-RL algorithm. A meta-RL algorithm is consistent if, given enough diverse data on the test task, adaptation can find a good policy for the task regardless of the training task distribution. Such an algorithm would provide a) rapid adaptation to new tasks from the same distribution as the train tasks while b) allowing for improvement even for out of distribution test tasks.

2. RELATED WORK

Meta-Reinforcement Learning is capable of learning new tasks with small amount of experience compared to Deep-RL which is heavily dependent on data. In the paper "Prefrontal cortex as a meta-reinforcement learning system", the author discusses how not just humans can learn a task by rewarding themselves but also learn new tasks quickly given prior experience in a similar task.

This paper builds on the idea of batch off-policy or offline reinforcement learning ([2]; [3]; [4]; [5]; [6]), extending the problem setting to the meta-learning setting. Current methods rely heavily on on-policy experience, limiting their sample efficiency. They also lack mechanisms to reason about task uncertainty when adapting to new tasks, limiting their effectiveness in sparse reward problems. In the paper "Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables"[7], the above challenges are addressed by developing an off-policy meta-RL algorithm, called Probabilistic Embeddings for Actor-critic RL (PEARL) which uses online probabilistic filtering of latent task variables to infer how to solve a new task from small amounts of experience.

3. BACKGROUND

3.1. MODEL AGNOSTIC META LEARNING(MAML)

[8]MAML, or Model-Agnostic Meta-Learning, is a model and task-agnostic algorithm for meta-learning that trains a model's parameters such that a small number of gradient updates will lead to fast learning on a new task. Consider a model represented by a parametrized function f_θ with parameters θ. When adapting to a new task T_i, the model's parameters θ become $\theta_i^{'}$. With MAML, the updated parameter vector $\theta_i^{'}$ is computed using one or more gradient descent updates on task T_i. For example, when using one gradient update, $\theta_i^{'} = \theta - \alpha \nabla_\theta L_{T_i}(f_\theta)$. The step size α may be fixed as a hyper parameter or meta-learned. The model parameters are trained by optimizing for the performance of $f_{\theta_i^{'}}$ with respect to θ across tasks sampled from $p(T_i)$. More concretely the meta-objective is as follows:

$$\min_\theta \sum_{T_i \sim p(T)} L_{T_i}(f_{\theta_i^{'}}) = \sum_{T_i \sim p(T)} L_{T_i}(f_\theta - \alpha \nabla_\theta L_{T_i}(f_\theta)).$$

The meta-optimization is performed over the model parameters θ, whereas the objective is computed using the updated model parameters $\theta^{'}$. In effect MAML aims to optimize the model parameters such that one or a small number of gradient steps on a new task will produce maximally effective behavior on that task. The meta-optimization across tasks is performed via stochastic gradient descent(SGD), such that the model parameters are updated as,

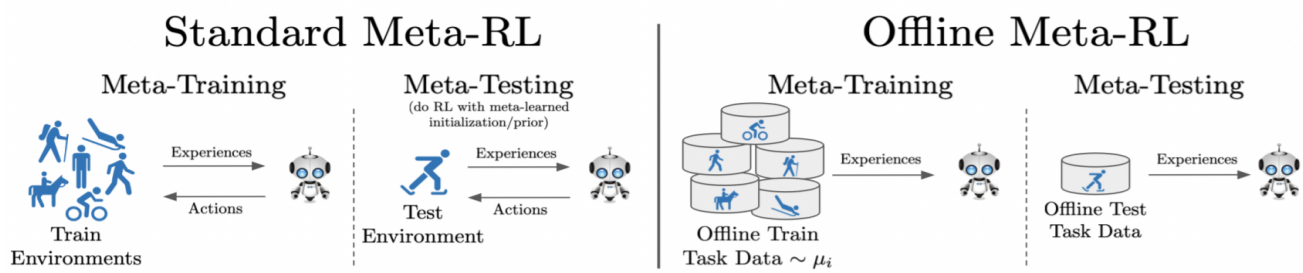

Fig. 1. Comparing the standard meta-RL setting (left), which includes on-policy and off-policy meta-RL, with offline meta-RL (right). In standard meta-RL, new interactions are sampled from the environment during both meta-training and meta-testing, potentially storing experiences in a replay buffer (off-policy meta-RL). In offline meta-RL, a batch of data is provided for each training task Ti.

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T_i)} L_{T_i}(f_{\theta_i'})$$

where β is the meta step size.

3.2. Advantage-weighted regression(AWR)

Advantage Weighted Regresion[9] is a simple off-policy algorithm for model-free RL. Each iteration of the AWR algorithm simply consists of two supervised regression steps: one for training a value function baseline via regression onto cumulative rewards, and another for training the policy via weighted regression.

3.3. PEARL

Probabilistic Embeddings for Actor-critic meta-RL[7] proposes an off-policy meta-RL algorithm. This algorithm enables fast adaptation by accumulating experience online, and performs structured exploration by reasoning about uncertainty over tasks.

4. MODEL AND METHOD

4.1. MACAW

A meta actor-critic with advantage weighting(MACAW) is proposed in this paper. MACAW is an offline meta-RL that learns the initializations ϕ and θ for a value function V_ϕ and policy π_θ respectively. It can rapidly adapt to a new task seen at meta-test time via gradient descent. Both the value function and the policy objectives correspond to simple weighted regression losses in both the inner and outer loop, leading to a stable and consistent inner-loop adaptation process and outer-loop meta-training signal. This is compared to the naive approach of using the AWR update in the inner loop.

In the inner loop procedure of MACAW the adaptation process consists of a value function update followed by a policy update. MACAW uses truncated optimization and bootstrap free update for the value function that simply performs supervised regression onto Monte-Carlo returns. Consider a batch of training data D_i^{tr} collected for T_i, MACAW adapts the value function by taking one or a few gradient steps on the following supervised objective:

$$\phi_i \leftarrow \phi - \eta_1 \nabla_\phi L_V(\phi, D_i^{tr}),$$

where $L_V(\phi, D) \underset{=}{\Delta} E_{s,a\sim D}[(V_\phi(s) R_D(s,a))^2]$ and $R_D(s,a)$ is the Monte Carlo return from the state s taking action a observed in D. After adapting the value function, we proceed to adapting the policy. While adapting the policy we can simply make use of the AWR algorithm where the advantage is given by $R_D(s,a) - V_{\phi_i'}(s)$, but we can observe that this update does not provide the meta learner with sufficient expressive power to be a universal update procedure for the policy [10] and for MAML-based methods to approximate any learning procedure the inner gradient must not discard information needed to infer the task [10] where as the gradient in AWR objective does not contain full information of both the regression weight and the regression target. To address this issue and make our meta-learner sufficiently expressive, an enriched policy update that performs both advantage-weighted regression onto actions as well as an additional regression onto action advantages is used in the MACAW algorithm. The MACAW policy architecture is shown in Fig.2.

This enriched policy update is only used during adaptation, and the predicted advantage is used only to enrich the inner loop policy update during meta-training; during meta-test, this predicted advantage is discarded. The enriched policy update proceeds as:

$$\theta_i \leftarrow \theta \alpha_1 \nabla_\theta L_\pi(\theta, \phi_i', D_i^{tr}), \text{where} L_\pi = L^{AWR} + L^{ADV}.$$

The AWR Loss and advantage regression loss is given by:

$$L^{ADV}(\theta, \phi_i', D) \underset{= s,a\sim D}{\Delta} E[(A_\theta(s,a)(R_D(s,a) V_{\phi_i'}(s)))^2],$$

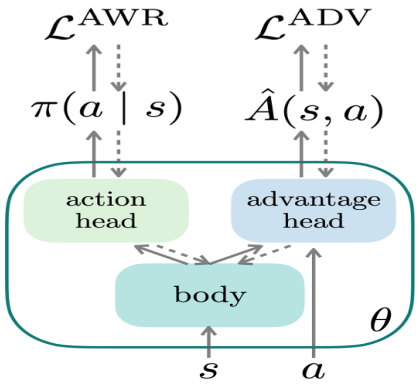

Fig. 2. MACAW policy architecture. Solid lines show forward pass; dashed lines show gradient flow during backward pass during adaptation only; the advantage head is not used in the outer loop policy update.

$$L^{AWR}(\vartheta, \varphi, B) = E_{s,a\sim B}[log\pi_\vartheta(a|s)exp(\frac{1}{T}(R_B(s,a)V_\varphi(s$$

Next, we describe the meta-training procedure for learning the meta-parameters and , the initialization of the policy and value function, respectively. In the outer loop MACAW procedure a batch of data D_i^{ts} is taken that is disjoint from the adaptation data D_i^{tr}. The meta-learning procedure for the value function is:

$$\min_\phi E_{T_i}[L_V(\phi_i^{'}, D_i^{ts})] = \min_\phi E_{T_i}[L_V(\phi - \eta_1 \nabla_\phi L_V(\phi, D_i^{tri}), D_i^{ts})].$$

Unlike the inner loop, we optimize the initial policy parameters in the outer loop with a standard advantage-weighted regression objective, since expressiveness concerns only pertain to the inner loop where only a small number of gradient steps are taken. Hence, the meta-objective for the initial policy parameters is:

$$\min_\theta E_{T_i}[L^{AWR}(\theta_i^{'}, \phi_i^{'}, D_i^{ts})] = \min_\phi E_{T_i}[L^{AWR}(\theta - \alpha_1 \nabla_\theta L_\pi(\theta, \phi_i^{'}, D_i^{tr}), \phi_i^{'}, D_i^{ts})].$$

5. RESULTS

In this paper we compare the performances of the proposed MACAW algorithm with state-of-art methods for offline Reinforcement Learning algorithm - Multitask AWR[9] and off policy metaRL algorithm[7]. Further we evaluate few variants of MACAW model i) MACAW's ability to leverage online fine-tuning at meta-test time ii) Importance of weight transform layer and improving the expressive power of MACAW. In this section we will look at data collection information and show some results.

5.1. Data Collection

In our experiment we mainly focus on 2 data sets:

1) Cheetah-direction: Train a cheetah to run in forward and backward directions. There are no held out test tasks here but it is uses as a proof of concept to prove the feasibility of this algorithm

2) Cheetah-velocity: Train a cheetah to run with some desired velocity. A subset of data is held out to test the model.

The same model can be extended to other datasets like Ant-2D direction, Walker-2D Params, MetaWorld etc.,

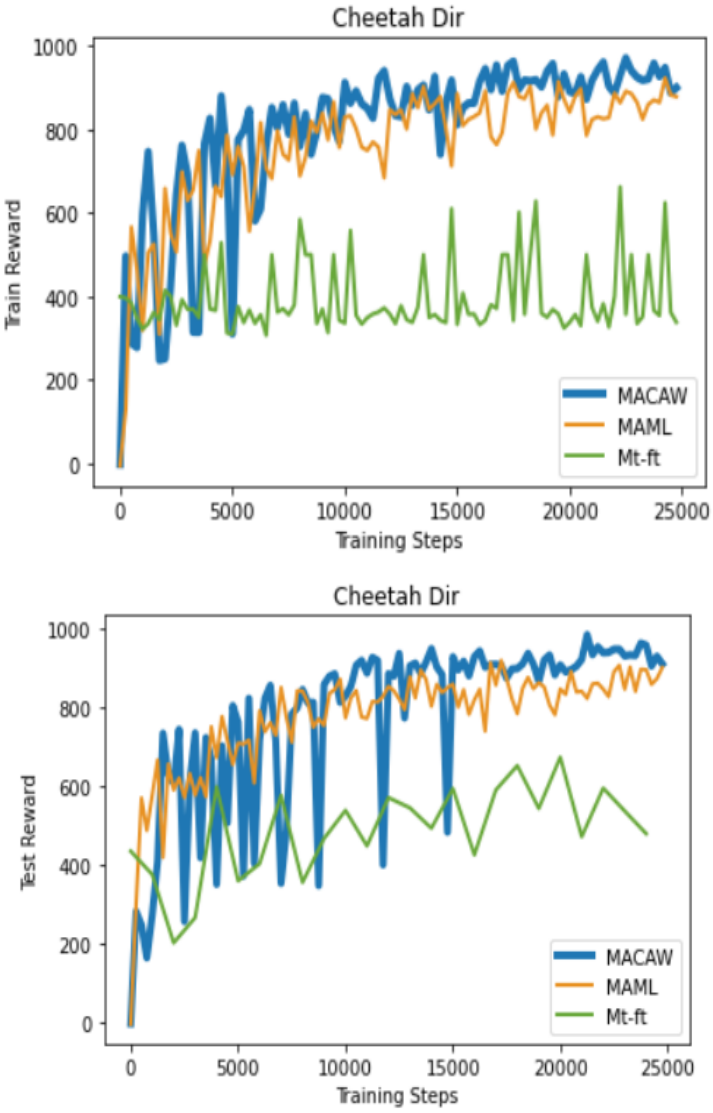

Fig. 3. Comparing the results of MACAW, MAML+AWR and multitask AWR on cheetah direction dataset.

5.2. Comparison of MACAW with other models

We test whether purely offline data is sufficient to train the model and adapt to new tasks. Here we compare MACAW with off-shelf offline RL models and meta learning models, specifically i) Multitask AWR[9] which consists of two standard supervised learning steps: one to regress onto target values for a value function, and another to regress onto weighted target actions for the policy. ii) MAML+AWR[8] combines Model-agnostic meta-learning, a meta-learning approach helps in faster adaptation with Advantage-Weighted Regression.

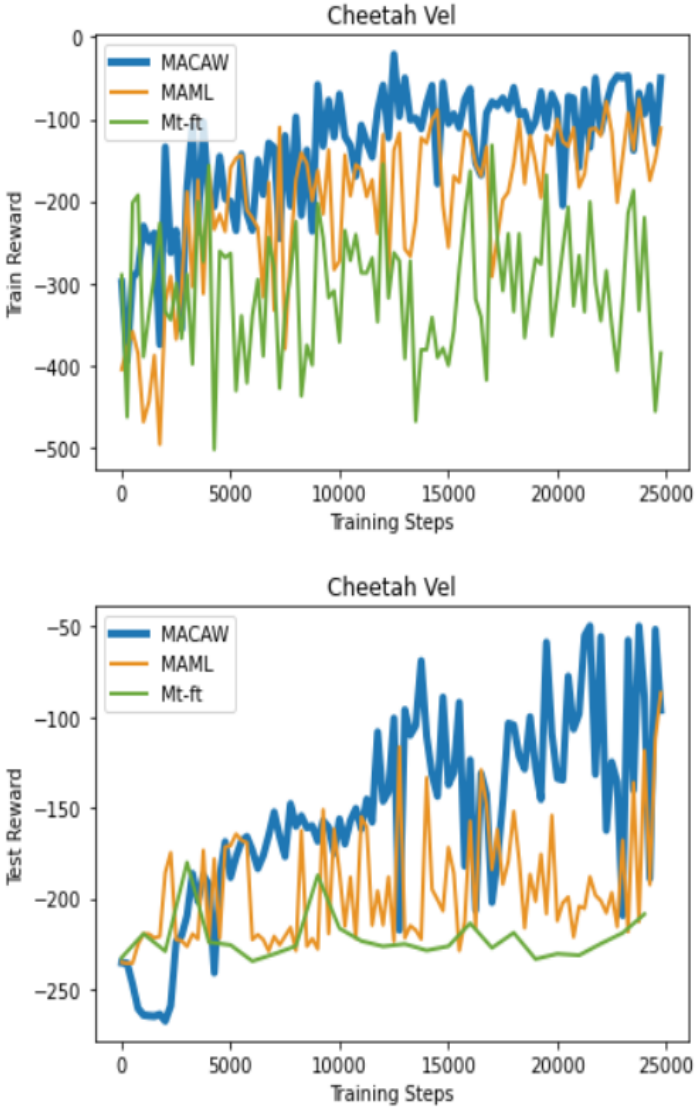

Fig. 4. Comparing the results of MACAW, MAML+AWR and multitask AWR on cheetah velocity dataset.

Results are shown in Fig 3 and Fig 4 for cheetah-dir and cheetah-vel respectively. MACAW outperforms all other state-of-art methods, while it has a significant improvement when compared with Multitask AWR + Fine-tuning, we can only see a slight improvement between MAML+AWR and MACAW models. Multitask AWR performs better on a simple dataset task like cheetah-direction but fails utterly on complex task like cheetah-velocity. MACAW is quite stable in training rewards but it is unstable during testing time, this may be attributed to training it on fewer training steps(25000) compared to the paper(500000).

5.3. Weight Transform Ablation Study

Here we try to understand the role of weight transform layer that increases the expressiveness of the MAML update. Past work by[11] shows that adding a bias transformation layer to each layer improves the expressiveness of MAML update. We extend this idea of computing the bias from a latent vector to the weight matrix.

$$w = W^{wt}z, \text{where} W^{wt} \in R^{(d^2+d)xc}$$

where first d^2 components is the weight matrix and next d components is the bias. Comparison of results with and without weight transformation layer can be show in Fig 6 for both cheetah-direction and cheetah-velocity dataset. We can clearly show that adding weight transform layers increase the reward significantly and also increased the stability of the model.

5.4. Varying the number of training tasks

The main goal of this paper is to adapt to new tasks even with minimum training data. Here we vary number of training tasks(half, quarter, 1/eighth, 1/sixteenth) and compare the performance of MACAW with offline multitask AWR + Fine-tuning model. Results on cheetah velocity data set are presented in the Fig 5.

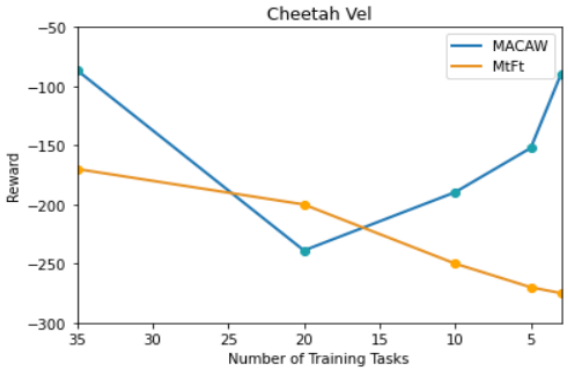

Fig. 5. Number of training tasks are varied and performance of MACAW and Multitask AWR is compared. It shows MACAW performs better that Multitask AWR even with less number of training tasks

MACAW shows most consistent performance even on very low number of training tasks, where as the performance of Multitask AWR gradually decreases as number of tasks decrease. This is a desirable property which proves efficiency of MACAW model.

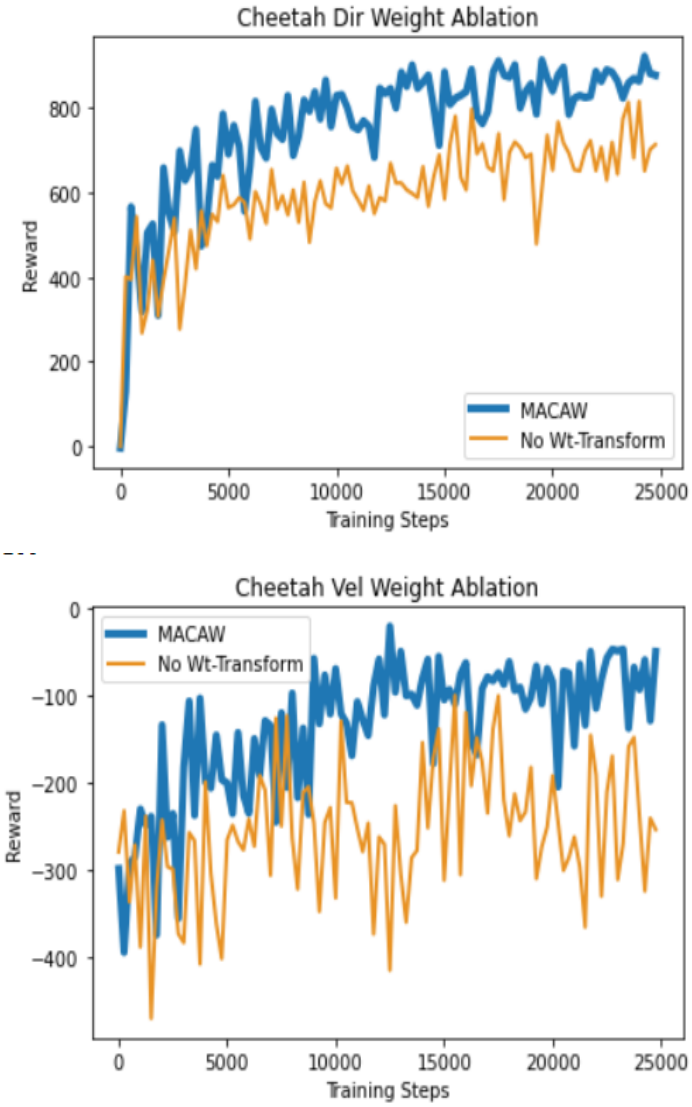

Fig. 6. Comparing performance of MACAW with and without weight transform layers for cheetah-dir and cheetah-vel datasets. MACAW performs better when a weight transformation layer is added in both the datasets.

6. CONCLUSION

In this paper we present macaw algorithm, that achieves better performance compared to state-of-art methods on various tasks. One of the limitations of MACAW algorithm is that it is able to adapt to new tasks from offline data but does not learn an exploration policy from offline data. We propose an addition of weight transform layers which improves the models performance. We also show that MACAW can also be trained with very few number of training tasks which makes the model very efficient to adapt to new tasks.

7. REFERENCES

[1] Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn, "Offline meta-reinforcement learning with advantage weighting," 2021.

[2] Scott Fujimoto, David Meger, and Doina Precup, "Off-policy deep reinforcement learning without exploration," in *Proceedings of the 36th International Conference on Machine Learning*, Kamalika Chaudhuri and Ruslan Salakhutdinov, Eds. 09–15 Jun 2019, vol. 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062, PMLR.

[3] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," 2019.

[4] Yifan Wu, George Tucker, and Ofir Nachum, "Behavior regularized offline reinforcement learning," 2019.

[5] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," 2020.

[6] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi, "An optimistic perspective on offline reinforcement learning," 2020.

[7] Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine, "Efficient off-policy meta-reinforcement learning via probabilistic context variables," 2019.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," 2017.

[9] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine, "Advantage-weighted regression: Simple and scalable off-policy reinforcement learning," 2019.

[10] Chelsea Finn and Sergey Levine, "Meta-learning and universality: Deep representations and gradient descent can approximate any learning algorithm," 2018.

[11] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine, "One-shot visual imitation learning via meta-learning," 2017.