Two-Stage Model-Agnostic Meta-Learning With Noise Mechanism for One-Shot Imitation

ZIYE HU1, ZHONGXUE GAN1,2, WEI LI1,2, JAMES ZHIQING WEN2, DECHENG ZHOU2, AND XUSHENG WANG1, (Student Member, IEEE)

1 Academy for Engineer and Technology, Fudan University, Shanghai 200433, China
2 Jihua Laboratory, Foshan 528000, China

Corresponding authors: Zhongxue Gan (ganzhongxue@fudan.edu.cn) and Wei Li (fd_liwei@fudan.edu.cn)

This work was supported in part by the Jihua Laboratory of China through the Self-Programming Intelligent Robot Project under Grant X190101TB190, in part by the Forward-Looking Research Project on the Basic Theory and Key Technology of Ternary-Space Swarm Intelligence under Grant 19511132000, in part by the Shanghai Engineering Research Center of AI and Robotics, and in part by the Engineering Research Center of AI and Robotics, Ministry of Education, China.

ABSTRACT

Given that humans and animals can learn new behaviors in a short time by observing others, the question we need to consider is how to make robots behave like humans or animals, that is, through effective demonstration, robots can quickly understand and learn a new ability. One possible solution is imitation based meta-learning, but most of the related approaches are limited in a particular network structure or a specific task. Particularly, meta-learning methods based on gradient-update are prone to overfit. In this article, we propose a generic meta-learning algorithm that divides the learning process into two independent stages (skill cloning and skill transfer) with a noise mechanism which is compatible with any model. The skill cloning stage enables a good understanding of the demonstration, which helps the skill transfer stage when the robot applies the learned experience into new tasks. The experimental results show that our algorithm can alleviate the phenomenon of overfitting by introducing a noise mechanism. Our method not only performs well on the regression task but is significantly better than the existing state-of-the-art one-shot imitation learning methods in the same simulation environments (i.e., simulated pushing and simulated reaching).

INDEX TERMS

Learning to learn, meta-learning, one-shot imitation learning, robot learning.

I. INTRODUCTION

A. MOTIVATION

As we know, humans and animals can learn new behaviors quickly by observing or imitating others and can effectively adapt to new environmental changes by using previous knowledge and experience. We expect that artificial agents can learn as fast as humans. Generally speaking, machine learning requires a large number of samples for training, while humans need only a small amount of samples to learn new skills and concepts. For example, humans only need to learn a few examples of cats and dogs to know the differences between their shapes and characteristics, so they can learn to distinguish between cats and dogs. Since the application environment of robots has migrated from simple settings to unstructured and complex environments, it requires a large amount of expert knowledge. And the corresponding programming has also become complicated, time-consuming, and expensive [1], [2]. Generally, we want robots to be adaptable as humans, which is almost impossible to achieve through traditional programming. Through demonstrations, we can unambiguously communicate any manipulation task, and simultaneously provide clues about the specific motor skills required by robots to perform the task [3]. As such, the core problem of meta-learning is how to build a machine learning model or method that can quickly learn from a small number of samples.

B. LITERATURE REVIEW

In imitation learning, one of the leading methods is behavior cloning based on supervised learning (e.g., [4]). However, when faced with a changing environment, methods based on behavioral cloning are often not adaptable and prone to overfitting. Unlike behavior cloning, the main idea of reinforcement learning [5], [6] is to acquire skills through a lot of trial and error, which has achieved remarkable success in many fields, such as continuous control in simulation or real environments [7]–[10], AlphaGo Zero [11], and Quake III Arena [12]. However, these methods based on reinforcement learning often access unsafe or undefined states space during training because of the need to interact with the environment.
and random behavioral exploration, while still not effectively solving the problem of quickly adapting to a new environment. In contrast to reinforcement learning, inverse reinforcement learning [13] selects the optimal behavior by learning a reward function [14]–[16] as an estimate, but requires additional expert knowledge to optimize rewards [16]. In general, reinforcement learning through random trial and error is a time-consuming task.

Although previous work has produced many impressive achievements in robotics, they have primarily considered each skill separately, rather than learning one skill to speed up learning another. Based on this background, meta-learning has been proposed in various concepts or forms [17]–[19] and is successfully applied to generative modeling [20], image recognition [21]–[25] and weight optimization algorithms [26]–[28]. However, these one-shot or few-shot meta-learning algorithms have not been applied to imitation learning. In the past, most operations of imitation learning (learning from demonstrations) methods were performed at the level of configuration-space trajectories [29]. These trajectories are usually collected by teleoperation [30], kinesthetic teaching [31], or sensors [32].

Unlike traditional methods based on manual programming, Duan et al. [3] proposed an excellent meta-learning framework that attempts to make the robot learn from very few demonstrations of any given task and instantly generalize to new situations without requiring task-specific engineering. Similarly, James et al. [33] introduced Task-Embedded Control Networks to leverage a task embedding to learn new tasks from demonstrations with meaningful results. Furthermore, Shao et al. [34] proposed a significant method for one-shot imitation combined with object detection, which is not an end-to-end framework. As it needs to train a carefully designed autoencoder network, a carefully designed object detection network, a carefully designed motion policy network, and it needs more manual labeling information, this method is very complicated and hard to train. Since these methods are mainly focusing on a specific network structure, it may lack versatility and flexibility for other tasks.

Not aiming at a particular network, Finn et al. [35] proposed a model-agnostic meta-learning algorithm (MAML), which is simple, elegant, and powerful to various tasks. Different from previous meta-learning methods [14], [25]–[27] for learning update functions or learning rules, the MAML algorithm neither expands the number of learning parameters nor constrains model architecture (e.g., by requiring a recurrent model [36] or a Siamese network [37]). As shown in Fig. 1 (left), the main idea of MAML is to make the model have the best performance on new tasks after updating the parameters through one or a few gradient steps. However, it is commonly thought that simple gradient descent would get trapped in terrible local minima [38] (see Fig. 2).

Based on MAML, Finn et al. [39], [40] proposed some significant work of one-shot visual imitation learning. According to the provided demonstration, one-shot imitation’s ideal goal is to make robots learn to deduce actions in a new environment. However, unlike image classification and regression tasks that need to provide labeled data during training and testing, imitation learning is more complicated. It is not easy to offer real-time actions corresponding to the demonstration in practical application. In other words, we tend to let the model directly infer the corresponding behaviors in a new scene without additional information from teaching videos. Therefore, it is tricky to make the model deduce works from a series of observations to new scenarios.

As shown in Fig. 1, the MAML algorithm for one-shot imitation (see Algorithm 1) tries to directly adopt the output of support tasks as a loss function to perform an internal gradient update to update model \( \theta \) since expert actions are not provided to the inner gradient update. Because MAML’s internal gradient update does not provide supervision information for inner loss, the inner loss is unpredictable (we found that the internal loss is quite huge in our experiments). Therefore, it cannot guarantee that the model has fully understood the behaviors of demonstrations (support tasks). After being updated by inner gradient descent, the model is directly enforced to adapt new scenes (target tasks). Intuition tells us that this learning model is more like seeking a direct mapping relationship from pictures to pictures than letting the model truly understand the meaning of pictures.

C. PAPER CONTRIBUTIONS

In order to solve the above problems, the contributions of our work are as follows:

- We propose a general two-stage model-agnostic meta-learning algorithm (TMAML, see Fig. 1 and Algorithm 2) for one-shot imitation. Compared to the MAML algorithm, TMAML divides the learning process into two independent stages:
  - skill cloning. In this stage, we try to let the model fully understand the intent of the demonstrations (support tasks) only;
  - skill transfer. In this stage, we transfer the learned knowledge (learned from support tasks) to the new scenes (target tasks).
* It should be noted that the TMAML is not simply adding more inner steps to MAML since it has its own outer gradient update step, and each stage of TMAML is relatively independent. In other words, we could consider skill cloning as pre-cognition or pre-training of skill transfer.

- In the process of meta-learning, we introduce a noise mechanism (see Algorithm 3) to alleviate the overfitting problem.
  - It should be noted that our noise mechanism is different from the work that is based on a complex probabilistic model proposed by Finn et al. [41]. In our work, we simply inject a global noise to the model \( \theta \) at the beginning of the inner gradient update, which is easy to implement and can increase the robustness of the model’s adaptability.
Z. Hu et al.: Two-Stage Model-Agnostic Meta-Learning With Noise Mechanism for One-Shot Imitation

**FIGURE 1.** Diagrams of our two-stage model-agnostic meta-learning algorithm (TMAML) and the MAML algorithm when we do not provide expert actions to inner gradient update. (a) Firstly, the MAML algorithm tries to directly adopt the output of support tasks as a loss function to perform an internal gradient update to update model $\theta$ since expert actions are not provided to the inner gradient update. Then the updated model $\theta^*$ is applied to adapt to new environments (target tasks). (b) Our TMAML algorithm attempts to divide the learning process into two stages: skill cloning and skill transfer. The purpose is to ensure that it has mastered the skills or knowledge from the given demonstrations (support tasks) and then apply the learned experience into new tasks (target tasks). In the skill cloning stage, we use the same internal gradient update strategy as well as MAML, but we do not directly apply the updated model to a new environment. In contrast, we provide expert actions for support tasks in the outer loss to make the model understand the actions required to complete according to the demonstrations from support tasks. After that, we assume the model has already grasped skills learned from support tasks, and we turn it into the skill transfer. In the skill transfer stage, we keep the same inner gradient update as before, and we perform a skill transfer at the outer gradient update to finish new tasks (target tasks) based on experience from support tasks. Note that we do only provide expert actions to the outer gradient update in both skill cloning or skill transfer.

**FIGURE 2.** Diagrams of simple gradient descent and our gradient descent with noise mechanism. (a) The simple gradient descent is easy to get trapped in poor local minima. (b) Suppose there is a small ball on the hill that gets trapped in poor local minima. If we gently push it, it may get rid of the previous local optimal point and fall into the global optimal point.

- In our work, we successfully apply our algorithm to the regression tasks, simulated pushing tasks, and simulated reaching tasks. We demonstrate that our algorithm achieved state-of-the-art performance in the same experimental setting compared to the previous advanced methods.

**II. PROBLEM FORMULATION OF META-IMITATION LEARNING**

For one-shot imitation, the MAML algorithm aims to train a model that can achieve rapid adaptation (a vision-based policy needed to adapt to a new scene from a single demonstration). In this section, we will define the visual meta-imitation learning problem and present the algorithm’s general form.

**A. PROBLEM STATEMENT**

Now we consider a model, indicated by $f$. This model’s function is to map demonstrations or observations $o$ to corresponding outputs or actions $\hat{a}$. Because we aim to apply our algorithm to various meta-learning tasks, we intend to introduce the same generic notion of a learning task below for convenience.

Formally, we denote each imitation task $\mathcal{T}_i = \{\tau = (o_1, a_1, \ldots, o_H, a_H) \sim \pi^*_i, q(o_{t+1}|o_t, a_t), C(a_{t+1}, a_t, H)\}$, where $\tau$ is demonstration data generated by an expert.
Algorithm 1 Model-Agnostic Meta-Learning (MAML) for One-Shot Imitation (differences in blue)

Require: \( p(T) \): distribution over tasks

Require: \( \alpha, \beta \): step size hyperparameters

1: randomly initialize \( \theta \)

2: while not done do

3: Sample batch of tasks \( T_i \sim p(T) \)

4: Divide the batch of tasks \( T_i \sim p(T) \) into two mini-batches: \( A_i \) as target tasks, \( B_i \) as target tasks \( (A_i \neq B_i) \)

5: for all \( A_i \) and \( B_i \) do

6: Evaluate \( L_{inner}(f_\theta) \) with respect to \( |A_i| \) examples:

7: \[ L_{inner}(f_\theta) = \frac{1}{|A_i|} \sum_{t \sim A_i} \| f_\theta(o_t) - a_t \|^2 \]

8: Compute adapted parameters with gradient descent:

9: \[ \theta'_i = \theta - \alpha \nabla_{\theta} L_{inner}(f_\theta) \]

10: Evaluate \( L_{outer}(f_\theta') \) with respect to \( |B_i| \) examples:

11: \[ L_{outer}(f_\theta') = \frac{1}{|B_i|} \sum_{t \sim B_i} \| f_\theta'(o_t) - a_t \|^2 \]

12: Update \( \theta \leftarrow \theta - \beta \nabla_{\theta} \sum_{B_i \sim p(T)} L_{outer}(f_\theta') \)

13: end for

14: end while

In the K-shot learning setting, we draw \( K \) samples of task \( T_i \sim p(T) \) as demonstrations for the model to train. For the one-shot imitation, the model needs to learn a new task \( T_i \) drawn from \( p(T) \) from only one demonstration generated by \( T_i \). Regarding meta-training, the model is trained using one demonstration from expert policy \( \pi^*_i \) based on a random task \( T_i \) sampled from \( p(T) \), and we test it on a new scene drawn from \( \pi^*_i \) to get the test error. The learned policy \( \pi_i \) is improved by optimizing the test error concerning the model parameters. Therefore, the test error serves as a training error of the meta-training process.

B. MODEL-AGNOSTIC META-LEARNING

In the meta-learning field, the MAML has been successfully applied to various scenarios, such as regression, image recognition, and reinforcement learning. The MAML algorithm tries to learn a model represented by weights \( \theta \) such that it can utilize standard gradient descent to get fast adaptation on new tasks \( T_i \) drawn from \( p(T) \). Since the algorithm takes the gradient descent as an inner optimizer, it does not need to introduce additional parameters, which is more parameter-efficient than other meta-learning methods. After learning from a demonstration, the model’s parameters \( \theta \) are updated as \( \theta'_i \) to adapt to a new task \( T_i \). Especially, the updated \( \theta'_i \) is computing through one or more gradient descent. For convenience, we mainly consider the case of a gradient update in the following sections, while multiple gradient updates could be seen as an extension. During training, the model’s parameters are optimizing according to the test error of \( f_\theta \):

\[
\min_{\theta} \sum_{T_i \sim p(T)} L_T(f_\theta) = \sum_{T_i \sim p(T)} L_T(f_\theta - \alpha \nabla_{\theta} L_T(f_\theta)).
\] (1)

where the hyperparameter \( \sigma \) is a step size in the inner gradient descent of meta-learning. Note that the meta-optimization is performed through inner gradient descent over parameters \( \theta \),
Algorithm 3 Meta-Learning Based on Noise Mechanism (differences From MAML in blue)

Require: \( p(T) \): distribution over tasks

Require: \( \alpha, \beta, \sigma \): step size hyperparameters

1: randomly initialize \( \theta \)

2: while not done do

3: Sample batch of tasks \( T_i \sim p(T) \) and random vector \( g_i \sim N(0, I) \)

4: Divide the batch of tasks \( T_i \sim p(T) \) into two mini-batches: \( A_i \) as target tasks, \( B_i \) as target tasks \((A_i \neq B_i)\)

5: for all \( A_i \) and \( B_i \) do

6: Compute parameters with noise vector:

\[
\theta^* = \theta - \sigma \cdot g_i
\]

7: Evaluate \( L_{inner}(f_{i^*}) \) with respect to \( |A_i| \) examples:

\[
L_{inner}(f_{i^*}) = \sum_{i \sim A_i} \left\| f_{i^*}(o_{i_j}(j)) \right\|_2^2 \]

8: Compute adapted parameters with gradient descent:

\[
\theta^*_i = \theta^* - \alpha \nabla_{\theta^*} L_{inner}(f_{i^*}) \]

9: Evaluate \( L_{outer}(f_{i^*_j}) \) with respect to \( |B_i| \) examples:

\[
L_{outer}(f_{i^*_j}) = \sum_{i \sim B_i} \sum_{t \sim T_i} \left\| f_{i^*_j}(o_{i^*_j}(j)) - a_{i^*_j}(j) \right\|_2^2 \]

10: Update \( \theta \leftarrow \theta - \beta \nabla_\theta \sum_{i \sim p(T)} L_{outer}(f_{i^*_j}) \)

11: end for

12: end while

Algorithm 2}

and it uses the updated \( \theta^*_i \) to produce results on new tasks. The test error of tasks is optimized by stochastic gradient descent (SGD).

C. MODEL-AGNOSTIC META-LEARNING FOR ONE-SHOT IMITATION

In this section, we will detail the extension of model-agnostic meta-learning algorithms (MAML) to imitation learning setting. Please note that we only provide visual information as a demonstration (support task). No other action and state information are included in an inner gradient update because they are tricky to collect in practical applications.

We use \( o \) to represent the input (e.g., a video) of the model, \( o_t \) represents the agent’s observation at time \( t \) (e.g., an image), and \( \hat{a} = f_0(o_t) \) indicates the predicted output (e.g., torques) at time \( t \). For simplicity, we denote a trajectory of demonstration as \( \tau = \{o_{1:T}, a_{1:T}\} \).

We assume that each task consists of at least two demonstrations for meta-learning. In meta-training, we randomly sample a batch of tasks \( T \), and each task consists of two demonstrations. For each task, we make a demonstration as the support task \((A_i)\) and another one as the target task \((B_i)\).

In the inner gradient update phase, we only provide visual information to the model without expert actions, and the inner loss is:

\[
L_{inner}(f_0) = \sum_{i \sim A_i} \sum_t \left\| f_0(o_{i_t}(j)) \right\|_2^2, \tag{2}
\]

where \( A_i \) is a minibatch of tasks drawn from \( T \sim p(T) \).

Although we do not provide expert actions in the inner gradient update for one-shot imitation, the expert actions are needed in the outer gradient update to get an objective loss. So in the outer gradient update phase, the expert actions are provided with visual information to the model, and the outer loss is:

\[
L_{outer}(f_{i^*_j}) = \sum_{i \sim B_i} \sum_t \left\| f_{i^*_j}(o_{i^*_j}(j)) - a_{i^*_j}(j) \right\|_2^2, \tag{3}
\]

where \( B_i \) is a minibatch of tasks drawn from \( T \sim p(T) \). We summarize the algorithm of MAML for one-shot imitation in Algorithm 3.

III. IMPROVED ALGORITHM

The MAML algorithm has achieved excellent results in regression, image classification, image super-resolution [42], etc. However, unlike these fields, behavior imitation based on vision is a very troublesome problem. For example, in the image classification task, we can easily provide the classification label for reference. Due to the randomness and unpredictability of human behavior, it is not easy to provide information (e.g., actions and states) other than visual information for the model in real-time without the assistance of additional devices. In terms of practical significance, humans can learn new behaviors only based on visual information, so we also hope that agents can complete tasks as efficiently as humans.

Since we tend to only provide visual information to the model in the inner gradient update phase according to the MAML algorithm, this will make the model lose supervision information as a reference during the intermediate learning process. In other words, the model is unable to fully understand what tasks the demonstrations are completing. On the contrary, we think this is a way to forcefully get the mapping relationship between support tasks and target tasks, rather than truly understanding. As a result, Table 1 lists the success rates of one-shot simulating pushing with varying demonstration information provided at test-time. And Table 2 lists the success rates of one-shot imitating reaching with different demonstration information provided at test-time. The results of these methods show that as the information provided in demonstrations (support tasks) decreases, the model’s cognitive ability declines sharply. Based on this, we hope to propose a method that can further understand the visual information from demonstrations and make the model have sufficient generalization capabilities.

A. TWO-STAGE MODEL-AGNOSTIC META-LEARNING ALGORITHM

As shown in Algorithm 2, we divide meta-learning in training into two stages: (i) skill cloning and (ii) skill transfer. As with the MAML algorithm, in order to ensure the generality of the algorithm, we do not make any assumptions about the form of the model but assume that a certain parameter vector \( \theta \) has parameterized it, and the loss function is smooth enough in \( \theta \) that we can use gradient-based methods. Now we use \( f_0 \) to represent the model function with respect to \( \theta \). At the
beginning of training, we need to randomly sample tasks $T_i \sim p(T)$ and divide them equally into two mini-batches: $A_i$ as support tasks and $B_i$ as target tasks. Different from MAML, we first perform skill cloning to let the model understand what actions to complete based on demonstrations. Because the model does not need to switch learning scenarios (from support tasks to target tasks) at this stage, it is easier to understand the purpose of tasks from demonstrations. An intuitive example is: We often practice some fundamental problems repeatedly instead of directly challenging new questions. After we have summarized the law of the problem, we will apply this law to a new similar topic, so that we will not feel confused.

1) SKILL CLONING

In the skill cloning stage, the model only learns from demonstrations $A_i \sim p(T)$ (which serve as both support tasks and target tasks) and performs internal gradient updates without supervision information (e.g., actions and states) for inner learning:

$$\theta'_i = \theta - \alpha \nabla \theta \mathcal{L}_{A_i}(f_\theta),$$

where the hyperparameter $\alpha$ stands for the step size of inner meta-learning. Here we mainly consider the case of one inner gradient update for convenience and directly adopt the internal output as a loss [40]. Still, more gradient updates could be used to produce better results. $\mathcal{L}_{A_i}(f_\theta)$ is:

$$\mathcal{L}_{inner}(f_\theta) = \mathcal{L}_{A_i}(f_\theta) = \sum_{i \sim A_i} \| f_\theta(\phi_i^{(j)}) \|^2_2.$$

After internal meta-learning, the model’s parameters $\theta$ are updated as $\theta'$. And then the performance of $f_{\theta'}$ with respect to $\theta$ is optimized via target tasks $A_i \sim p(T)$:

$$\min_{\theta} \sum_{A_i \sim p(T)} \mathcal{L}_{A_i}(f_{\theta'-\alpha \nabla \theta \mathcal{L}_{A_i}(f_\theta)}).$$

Note that the meta-objective is to finish the target task according to the support task, so we must provide supervision information for the outer loss $\mathcal{L}_{A_i}(f_{\theta'})$. More concretely, the $\mathcal{L}_{A_i}(f_{\theta'})$ is different from $\mathcal{L}_{A_i}(f_\theta)$ as:

$$\mathcal{L}_{outer}(f_{\theta'}) = \mathcal{L}_{A_i}(f_{\theta'}) = \sum_{i \sim A_i} \| f_{\theta'}(\phi_i^{(j)}) - \phi_i^{(j)} \|^2_2.$$

According to the external loss calculated by (7), the final step of skill cloning is to use stochastic gradient descent (SGD) to optimize the model parameter $\theta$:

$$\theta \leftarrow \theta - \beta \nabla \theta \sum_{A_i \sim p(T)} \mathcal{L}_{A_i}(f_{\theta'}),$$

where the hyperparameter $\beta$ stands for the meta step size of external meta-learning, and we call this step as an outer gradient update in this article.

Although we do not provide supervision information in an internal gradient update, we can offer it directly in the outer gradient update. Notably, the support tasks and target tasks are kept the same in the skill cloning stage. Since it does not need to transfer a scene to another scene during training in this phase, it will be more natural for the robots to understand the tasks without facing the ambiguity of different demonstrations (e.g., a pair of related videos A and B).

2) SKILL TRANSFER

After completing a skill cloning stage, we assume that the model has understood the demonstrations of support tasks, and we turn it to the skill transfer stage. In the skill transfer stage, the model not only learns from demonstrations $A_i \sim p(T)$ (which serve as support tasks) but also apply the learned information to the new environments $B_i \sim p(T)$ (which serve as target tasks).

In the internal gradient update, the skill transfer stage and the skill cloning stage are consistent:

$$\phi'_i = \theta - \gamma \nabla \theta \mathcal{L}_{A_i}(f_\theta),$$

where the hyperparameter $\gamma$ stands for the step size.

Because we have trained $A_i \sim p(T)$ independently in the skill cloning stage, we assume that the model has learned the corresponding skills in the internal gradient update, and it can be transferred to a new environment in the outer gradient update. Therefore, the meta-objective of skill transfer stage is optimizing the loss for the new tasks $B_i \sim p(T)$ different from $A_i \sim p(T)$:

$$\mathcal{L}_{outer}(f_{\phi'}) = \mathcal{L}_{B_i}(f_{\phi'}) = \sum_{i \sim B_i} \| f_{\phi'}(\phi_i^{(j)}) - \phi_i^{(j)} \|^2_2.$$

According to the loss calculated by the (10), the final step of skill transfer is performing the meta-optimization through stochastic gradient descent (SGG) to update the model parameter $\theta$:

$$\theta \leftarrow \theta - \delta \nabla \theta \sum_{B_i \sim p(T)} \mathcal{L}_{B_i}(f_{\phi'})$$

where the hyperparameters $\delta$ is a meta step size in skill transfer of meta-learning.
In the process of TMAML, we alternately perform skill cloning and skill transfer in each iteration (we do not provide supervision information for imitation learning in the internal updates of these two stages). The most significant difference between them is that the former stage updates model parameters $\theta$ only based on tasks $A_i$ sampled from $p(T)$. In this stage, the model can learn skills without ambiguity since there is no scene transition. And the latter stage updates model parameters $\theta$ based on both tasks $A_i$ and $B_i$ sampled from $p(T)$ with scene transition. In the following experiments, we will prove that skill cloning can promote the understanding of skill transfer.

Please note the TMAML is not simply adding more internal steps to MAML for one-shot imitation, since we train the skill cloning and skill transfer separately. Moreover, the support tasks’ supervision information of MAML in our one-shot imitation setting is never provided in the internal step during training. On the contrary, we cleverly offer it in the outer gradient update of the skill cloning stage to help the understanding of the demonstration.

B. META-LEARNING BASED ON NOISE MECHANISM

Since the MAML algorithm heavily relies on the gradient direction provided by demonstrations, the gradient direction’s accuracy becomes extremely important. However, as mentioned before, methods based on gradient updates often encounter overfitting problems, especially when facing new unseen tasks or demonstrations. For example, we first performed regression experiments based on MAML and found that after the gradient steps were increased to a certain number, the performance did not improve as we expect. We consider that the internal gradient updates have overfitted, especially in the case of simple tasks.

As such, we introduce a noise mechanism in meta-learning. As shown in Fig. 2, we suppose a small ball is on the hill that gets trapped in terrible local minima. If we apply an external force on the little ball, it may move and slide to the global minimum. Here we compare the parameter $\theta$ to be optimized to a small ball, and the noise is analogous to the external force exerted on this ball. More concretely, we add some random noise to the model’s parameters $\theta$ at the beginning of each internal gradient update during training:

$$\theta^* = \theta - \sigma \cdot g_i, \quad (12)$$

where $\sigma$ is a step size and $g_i$ could be some random Gaussian noise. Since we first update the model’s parameters $\theta$ according to some noise, we could alleviate the model’s overfitting phenomenon and increase its generalization ability.

Based on the updated $\theta^*$, we perform an internal gradient update:

$$\theta_i' = \theta^* - \alpha \nabla_{\theta^*} L_{A_i}(f_{\theta^*}). \quad (13)$$

We describe the specific algorithm in Algorithm 3.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

We mainly conduct experiments in three tasks: regression, simulated reaching, simulated pushing. We design these experiments to answer the following questions:

- What is the performance of TMAML compared with previously mentioned methods?
- What effect does skill cloning have on skill transfer in our TMAML algorithm?
- What is the performance of merely training the MAML algorithm twice compared to the skill cloning phase of our TMAML algorithm?
- How does the noise mechanism affect the performance of meta-learning?

In order to make our algorithm more convincing, we will compare it with some existing state-of-the-art experimental methods with the same problem settings and network structures.

A. REGRESSION

We first begin with a simple regression problem and compare it with MAML (see Fig. 5), and the experimental settings are consistent with [35]. In the regression task, we map each input $x$ to a specific sine wave output $f(x)$, where the amplitude and phase of the sinusoid are randomly sampled. Please note that the distribution of $p(T)$ is continuous, where the amplitude takes a value within $[0, 5.0]$ and the phase takes a value within $[0, \pi]$.

During meta-learning, input $x$ consists of $K$ data points, randomly sampled within $[-5.0, 5.0]$. We use mean-squared error (MSE) to evaluate the loss between the true output $y$ and the predicted output $f_\theta(x)$, where $f_\theta$ is a neural network regressor with two hidden layers of size 40 and the activation function is ReLU. In training, we use ten internal gradient updates with $K = 10$ examples and set Adam [43] as a meta-optimizer (see [35] for more details regarding training and setting about MAML). For TMAML, we set with $\alpha = \gamma = \delta = 0.001$ and $\beta = 10^{-6}$. Moreover, we also combine TMAML with noise mechanism and set the $\sigma$ in Algorithm 3 to 0.01 for training (0.0 for testing).

The experimental results (see Fig. 5 (b) and (c)) show that as the internal gradient step of the MAML algorithm increases, sometimes its performance is not improved but overfitting. In contrast, our “TMAML + noise” algorithm can predict regression well even when the provided demonstration used for meta-learning contains some inaccurate points. We show that our algorithm achieves significant performance, and conventional gradient-descent methods cannot adapt to changing regression problems without additional gradient update strategies. Moreover, we performed an ablation experiment on the MAML algorithm, which shows that merely increasing the training times with the same samples does not significantly improve the training performance, thereby confirming our algorithm’s effectiveness. In this regression experiment, we can conclude that our algorithm is superior to the previous methods. The introduction of the noise mechanism increases the robustness of the model and alleviates the overfitting phenomenon, and the skill cloning step has a promoting effect on the skill transfer.
FIGURE 3. Examples of simulated reaching. This experimental dataset comes from [39] and [40] based on MuJoCo and Gym environments, which is open-source on Github. In the simulated reaching problem, the robot should learn to reach a particular color learning from the demonstrations (support tasks) with a changeable target and environment.

FIGURE 4. Examples of simulated pushing. This experimental dataset is also provided by [39] and [40]. In the simulated pushing task, the robot should push the object to the specific place after watching a demonstration (support task) with a changeable goal in a different scene.

B. SIMULATED IMITATION

Unlike the previous regression problem, we hope our method can map image pixels to corresponding actions without labels or expert actions from demonstrations (support tasks). Since our algorithm is not aimed at a specific model structure, we adopt the excellent network structure proposed in [40] as an experimental model. Based on one-shot imitation learning, we mainly carry out experiments in simulated reaching and simulated pushing.

1) SIMULATED REACHING

As illustrated in Fig. 3, the goal of simulated reaching is to reach a particular color in the target environment after watching a demonstration (support task), which is disturbed by some distractors of different colors. In the simulated reaching problem, we set $\alpha = \gamma = \beta = \delta = 0.001$, batch_size = 25, training_iterations = 30000 and the number of convolution layers is 5 with $3 \times 3$ filters in the model proposed by [40] for TMAML. To combined TMAML with noise mechanism, we set $\sigma = 1e^{-10}$ for “MIL(temporal loss + noise)”, $\sigma = 1e^{-12}$ for “TMAML + noise” applied to skill transfer only, and we set the number of internal gradient updates to 1.

2) SIMULATED PUSHING

As illustrated in Fig. 4, the goal of simulated pushing is to push a particular object to a specific place after watching a demonstration (support task), which is disturbed by some distractors of different objects. In the simulated pushing problem, we set $\alpha = \gamma = 0.01, \beta = \delta = 0.001$, batch_size = 20, training_iterations = 30000 in the model proposed by [40] for TMAML. Regarding noise mechanism, we set $\sigma = 1e^{-10}$ for “MIL(temporal loss + noise)”, $\sigma = 1e^{-12}$ for “TMAML + noise” applied to skill transfer only, and we set the number of internal gradient updates to 1.

C. DATA ANALYSIS

Regarding one-shot imitation learning, we show the experimental results in Fig. 6 and Table 3. Now we will combine the data to answer the four questions we propose before as follows:

- **Question:** What is the performance of TMAML compared with previously mentioned methods?
  - **Answer:** As shown in Table 3, we get the accuracy of 83.33% in simulated pushing and 95.28% in simulated reaching, which are better than the current advanced methods.

- **Question:** What effect does skill cloning have on skill transfer in our TMAML algorithm?
  - **Answer:** As shown in Fig. 6, we record the training loss of the simulated reaching. We can find that the loss in the skill cloning stage drops very rapidly and gradually becomes flat, which indicates that the model has understood the content of demonstrations (support tasks) without much ambiguity. With the assistance of skill cloning, the loss in skill transfer drops fast and becomes relatively steady. However, the loss of other methods fluctuates greatly, which shows that it is difficult for the model to transfer learning between scenarios (from support tasks $A_i$ to target tasks $B_i$) directly.
FIGURE 5. Experimental results of the regression problem. (a) Comparison of different methods for the few-shot adaptation, where “No grad step” represents a standard gradient-descent method. (b) Comparison of the TMAML with noise mechanism and MAML, where “TMAML + noise” means ten regular internal gradient steps and ten internal gradient steps with noise mechanism, “used for grad” is the demonstration points used for meta-learning. (c) Mean-squared error (MSE) loss of different methods. Since the MAML algorithm does not have a skill cloning phase, in order to ensure that the parameters $\theta$ are updated with the same times, we set up an ablation experiment of “Ablation of MAML.” In the “Ablation of MAML” setting, we double the training times based on the same samples according to the MAML algorithm with ten inner gradient steps.

FIGURE 6. Training loss of the reaching task. Where Ablation of MIL (train twice) means training the samples twice in each iteration on the basis of MIL (with temporal loss) [40], which corresponds to our skill cloning and skill transfer respectively.

TABLE 3. The success rates of one-shot pushing and reaching (we use only visual information from demonstrations as support tasks).

| Methods                          | pushing | reaching |
|----------------------------------|---------|----------|
| LSTM [3, 43]                     | 34.23%  | –        |
| contextual [43]                  | 56.98%  | –        |
| MIL, linear loss [42, 43]        | 64.44%  | –        |
| Task-Embedded Control Networks [46] | 77.25% | 86.31%   |
| MIL, temporal loss [43]          | 80.63%  | 91.79%   |
| Ablation of MIL                  | 79.50%  | 84.23%   |
| MIL, temporal loss + noise(ours) | 81.76%  | 92.64%   |
| TMAML (ours)                     | 88.33%  | 95.28%   |
| TMAML + noise (ours)             | 82.88%  | 95.47%   |

- Question: What is the performance of merely training the MAML algorithm twice compared to the skill cloning phase of our TMAML algorithm?
  - Answer: As illustrated in Fig. 6, in terms of loss, merely training the MAML algorithm twice is not superior to TMAML. Combining the results in Table 3, we find that although the method has reduced the loss in training, the accuracy drops to 79.50% in the pushing task and 84.23% in the reaching task. We infer the approach is easy to overfit the training set, which produces poor experimental results in the testing set.

- Question: How does the noise mechanism affect the performance of meta-learning?
  - Answer: First, let us focus on Fig. 5 (c). We find that as the number of internal gradients increases, the MAML-based method’s loss does not decrease as we expected. However, with the introduction of noise mechanism, the overall loss of the model has been further improved. In Table 3, we find the accuracies are improved in both pushing and reaching of “MIL, temporal loss + noise(ours)”. As for “TMAML + noise(ours)”, the accuracy of reaching increases to 95.47%, but it drops to 82.88% in pushing task. According to these results, we infer that the noise mechanism can promote model performance and alleviate the gradient-based overfitting in some cases (i.e., simulated reaching and regression). However, it should be noted that sometimes random noise can also reduce performance, especially for tasks or environments that are particularly sensitive to noise. For example, simulated pushing is sensitive to any changes since we amplify the loss by 50 times for training.

V. CONCLUSION AND FUTURE WORK

In this study, we presented an effective meta-learning method that is universal to various tasks without specific models, which can quickly adapt to new and unseen scenarios based on demonstrations. We demonstrated the effectiveness of our approach to tasks of simulated pushing, simulated reaching, and regression with state-of-the-art results. The experimental results show that our method has a better understanding of visual information, which can effectively perform
generalization of knowledge and transfer to new application scenarios of the same task. We also introduced a noise mechanism for the overfitting problem, which can further improve model performance at a cheap cost.

There are many meaningful research directions in the future, such as cross-task experience sharing and knowledge reuse with a universal algorithm. We plan to extend the algorithm to apply it to multiskasking mechanisms. As we know, humans can handle a variety of tasks at the same time. Intuition tells us that there may be some common knowledge between different tasks that can be shared and reused quickly, and how to efficiently use the commonality between different tasks and avoid mutual interference of different tasks will be an essential topic in the future. Further and most importantly, we hope to explore how to distill the learning experience that was previously available and quickly apply it to entirely new and different unseen tasks.

REFERENCES

[1] A. Hussein, M. M. Gaber, E. Elayan, and C. Jayne, “Imitation learning: A survey of learning methods,” ACM Comput. Surv., vol. 50, no. 2, p. 21, 2017.

[2] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, “An algorithmic perspective on imitation learning,” Found. Trends Robot., vol. 7, nos. 1–2, pp. 1–179, 2018.

[3] Y. Duan, M. Andrychowicz, B. Stadie, O. J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, “One-shot imitation learning,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 1087–1098.

[4] D. Pomerleau, “An autonomous land vehicle in a neural network,” in Advances in Neural Information Processing Systems, vol. 1, San Mateo, CA, USA: Morgan Kaufmann, 1998.

[5] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press, 2018.

[6] L. P. Kaehbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” J. Artif. Intell. Res., vol. 4, pp. 237–285, May 1996.

[7] N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, and Y. Tassa, “Learning continuous control policies by stochastic value gradients,” in Proc. Adv. Neural Inf. Process. Syst., 2015, pp. 2944–2952.

[8] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” 2015, arXiv:1509.02971. [Online]. Available: http://arxiv.org/abs/1509.02971

[9] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, “Benchmarking deep reinforcement learning for continuous control,” in Proc. Int. Conf. Mach. Learn., 2016, pp. 1329–1338.

[10] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” J. Mach. Learn. Res., vol. 17, no. 1, pp. 1334–1373, 2016.

[11] D. Silver, J. Schrittwieser, I. Simonyan, D. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, “Mastering the game of go without human knowledge,” Nature, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[12] M. Jaderberg, W. M. Czarnecki, I. Dunning, L. Marris, G. Lever, D. Silver, and D. Wierstra, “Siamese neural networks for one-shot image recognition,” in Proc. ICML Deep Learn. Workshop, 2015, pp. 1–27.

[13] O. Vinyals, C. Blundell, T. Lillicrap, and D. Wierstra, “Matching networks for one shot learning,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 3630–3638.

[14] H. Edwards and A. Storkey, “Towards a neural statistician,” 2016, arXiv:1606.02185. [Online]. Available: http://arxiv.org/abs/1606.02185

[15] L. Kaiser, O. Nachum, A. Roy, and S. Bengio, “Learning to remember rare events,” 2017, arXiv:1703.03129. [Online]. Available: http://arxiv.org/abs/1703.03129

[16] S. Ravi and H. Larochelle, “Optimization as a model for few-shot learning,” in Proc. ICLR, Paris, France, 2017.

[17] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, “Learning to learn by gradient descent by gradient descent,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 3981–3989.

[18] S. Bengio, Y. Bengio, J. Cloutier, and J. Glecge, “On the optimization of a synaptic learning rule,” in Proc. Optimality.Artif. Biol. Neural. Netw., 1992, pp. 6–8.

[19] J. Schmidhuber, “Learning to control fast-weight memories: An alternative to dynamic recurrent networks,” Neural Comput., vol. 4, no. 1, pp. 131–139, Jan. 1992.

[20] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, “A survey of robot learning from demonstration,” Robot. Auto. Syst., vol. 57, no. 5, pp. 469–483, May 2009.

[21] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, “Deep imitation learning for complex manipulation tasks from virtual reality teleoperation,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2018, pp. 1–8.

[22] B. Akgun, M. Cakmak, J. W. Yoo, and A. L. Thomaz, “Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective,” in Proc. 7th Annu. ACM/IEEE Int. Conf. Hum.-Robot Interact., 2012, pp. 391–398.

[23] N. Ekvall and D. Kragic, “Interactive grasp learning based on human demonstration,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), vol. 4, Apr. 2004, pp. 3519–3524.

[24] S. James, M. Bloesch, and A. J. Davison, “Task-embedded control networks for few-shot imitation learning,” 2018, arXiv:1810.03237. [Online]. Available: http://arxiv.org/abs/1810.03237

[25] Q. Shao, J. Qi, J. Ma, Y. Fang, W. Wang, and J. Hu, “Object detection-based one-shot imitation learning with an RGB-D camera,” Appl. Sci., vol. 10, no. 3, p. 803, Jan. 2020.

[26] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in Proc. 34th Int. Conf. Mach. Learn., vol. 70, 2017, pp. 1126–1135.

[27] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, “Meta-learning with memory-augmented neural networks,” in Proc. Int. Conf. Learn. Represent., 2016, pp. 1842–1850.

[28] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, “Decaf: A deep convolutional activation feature for generic visual recognition,” in Proc. Int. Conf. Mach. Learn., 2014, pp. 647–655.

[29] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[30] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, “One-shot visual imitation learning via meta-learning,” 2017, arXiv:1709.04905. [Online]. Available: http://arxiv.org/abs/1709.04905

[31] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine, “One-shot imitation from observing humans via domain-adaptive meta-learning,” 2018, arXiv:1802.01557. [Online]. Available: http://arxiv.org/abs/1802.01557

[32] C. Finn, K. Xu, and S. Levine, “Probabilistic model-agnostic meta-learning,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 9516–9527.
ZIYE HU received the bachelor’s degree in the digital media technology specialty from the College of Computer Science and Electronic Engineering, Hunan University, China, in 2014. He is currently pursuing the Ph.D. degree in academy for engineer and technology from Fudan University, China, majored in computer application technology. His research interests include deep learning, neural networks, meta-learning, robot imitation learning, image processing, and so on. He received the awards such as the National Encouragement Scholarship, the Corporate Scholarship, the Outstanding Academic Scholarship, and so on.

ZHONGXUE GAN received the Ph.D. degree in mechanical engineering from the University of Connecticut, in 1993. He is currently an Expert of intelligent manufacturing and energy system engineering. In 1994, he joined the American Energy and Power Research Center, ABB Group. He served as a Project Manager, a Senior Researcher, the Research Center Director, and the Chief Scientist with the ABB Group. He has presided over the development of ABB IRB’s third and fourth generation robot controllers. He has invented the flexible, intelligent control technology, and he has completed various ABB controllers from motion controllers to force-vision hybrid controllers. He has a significant contribution to the transformation and upgrading of smart controllers based on behavioral intelligence so that ABB’s controller performance ranked first in the area of industrial robots. Additionally, he has also presided over one major demonstration project in China, two international scientific, and technological cooperation projects. He has his own 15 U.S. patents and 126 Chinese patents. He has published more than 40 articles, three books. His research interests include the underlying theory and critical technology of swarm intelligence, autonomous intelligent robot, and flexible automatic control.

Dr. Gan received the People’s Republic of China International Science and Technology Cooperation Award. He plays a vital role at Fudan University, where he is the President of the Institute of Intelligent Robotics and the Vice President of the Institute of Engineering and Applied Technology. Meanwhile, he is the President in the academy of the intelligent manufacturing industry and the Emergent Group in Ningbo of China.

WEI LI received the B.Eng. degree in automation and the M.Eng. degree in control science and engineering from the Harbin Institute of Technology, China, in 2009 and 2011, respectively, and the Ph.D. degree from the University of Sheffield, U.K., in 2016. He is currently an Associate Professor with the Institute of AI and Robotics, Fudan University. He has published more than 20 academic articles in peer-reviewed journals and conferences, such as the IEEE TRANSACTIONS ON ROBOTICS and NeurIPS. His research interests include robotics and computational intelligence, and specifically self-organized systems, and evolutionary machine learning.

JAMES ZHIQING WEN received the Ph.D. degree in optical instruments from Tsinghua University, Beijing, China, in 1994. From 1994 to 1995, he was a Postdoctoral Fellow of electrical and computer engineering with the University of California, Santa Barbara, USA. He then worked at several other organizations in USA. In May 2019, he joined the Jihua Laboratory, Foshan, China, where he serves as the Executive Deputy Director of the Engineering Research Center for Intelligent Robotics. His current research interests include electro-optics, 2D/3D imaging, computer vision, robotics, AI, VR, AR, medical imaging, automated inspection, and sensor fusion. He has authored and co-authored more than 60 scientific articles. He received the 2002 Rudolf Kingslake Medal and Prize of the International Society of Optical Engineers.

DECHENG ZHOU received the dual bachelor’s degree in mechanical engineering and computer application technology from Hunan University, Hunan, China, in 2001. He is currently a Senior Engineer. He was once the General Manager of the Huizhou Southern Intelligent Manufacturing Industry Research Institute, the Manager of the IE Department of Langshi Furniture Company Ltd., and Supply Chain Management Project Manager of Nortel Networks. In May 2019, he joined the Jihua Laboratory, Foshan, China, where he acts as the Team Leader of the Automatic Programming Intelligent Robotics in the Engineering Research Center for Intelligent Robotics. He has been involved in intelligent manufacturing project management-related work for a long time. He has a deep understanding and rich experience in enterprise intelligent manufacturing need, transformation of scientific and technological achievements, and project implementation management. His current research interests include robotics, machine vision, and integration of the two IT and OT.

XUSHENG WANG (Student Member, IEEE) received the master’s degree from the University of International Business and Economics. He is currently pursuing the Engineering degree majored in electronic information with Fudan University. He has been awarded the Honor of Outstanding Inventor in Hebei Province of China. He has participated in many national, provincial, and ministerial critical scientific research projects. He has published many academic articles and applied for 25 patents, including 19 invention patents, and ten authorized patents. His research interests include medical service robots and swarm intelligence.