Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Review:

Deep reinforcement learning: a survey*

Hao-nan WANG‡, Ning LIU, Yi-yun ZHANG, Da-wei FENG, Feng HUANG,
Dong-sheng LI, Yi-ming ZHANG

Science and Technology on Parallel and Distributed Processing Laboratory,
National University of Defense Technology, Changsha 410000, China

E-mail: wanghaonan14@nudt.edu.cn; liuning17a@nudt.edu.cn; zhangyiyun213@163.com; fengdawei@nudt.edu.cn;
huangfeng@nudt.edu.cn; dsli@nudt.edu.cn; zhangyiming@nudt.edu.cn

Received Sept. 29, 2019; Revision accepted Mar. 30, 2020; Crosschecked June 4, 2020

Abstract: Deep reinforcement learning (RL) has become one of the most popular topics in artificial intelligence research. It has been widely used in various fields, such as end-to-end control, robotic control, recommendation systems, and natural language dialogue systems. In this survey, we systematically categorize the deep RL algorithms and applications, and provide a detailed review over existing deep RL algorithms by dividing them into model-based methods, model-free methods, and advanced RL methods. We thoroughly analyze the advances including exploration, inverse RL, and transfer RL. Finally, we outline the current representative applications, and analyze four open problems for future research.

Key words: Reinforcement learning; Deep reinforcement learning; Reinforcement learning applications
https://doi.org/10.1631/FITEE.1900533 **CLC number:** TP18

1 Introduction

With the combination of deep learning and big data, revolutionary advances have occurred in artificial intelligence research. There is growing interest to explore new technologies in the field of post-deep learning. Deep reinforcement learning (RL), which uses neural network modeling in traditional RL algorithms, is particularly attractive. Specifically, deep RL is used to solve decision optimization problems, and decides which action to perform to maximize the benefit in the face of a specific state. As a result, both the academic community and industry are paying much attention to analyzing and applying deep RL.

Deep RL is a general paradigm which combines RL and deep learning and has achieved success in a variety of scenarios, such as chess, investment, driving, and action imitation. Deep RL is thought as one of the closest things that look anything like artificial general intelligence (AGI).

The processing and analysis differences between deep RL and traditional machine learning are huge. The current mainstream machine learning paradigm mostly collects or constructs dataset tags in advance, and performs machine learning based on existing static data. By contrast, RL is a typical representation of the closed-loop learning paradigm, which uses dynamic data and tags to bring feedback signals into the learning process. In this survey, we try to provide an overview of the stat-of-the-art deep RL algorithms.

Several comprehensive reviews have been published on deep RL. For example, Mousavi et al. (2018) summarized some early RL algorithms and categorized them into six core elements, six important mechanisms, and 12 applications. They listed

‡ Corresponding author
* Project supported by the National Natural Science Foundation of China (Nos. 61772541, 61872376, and 61932001)
ORCID: Hao-nan WANG, https://orcid.org/0000-0002-0792-3858

a collection of RL resources. Vanschoren (2018) reviewed meta-learning algorithms and categorized them based on the type of meta-data. Vanschoren (2018) gave experimental evaluation and analysis of the performance characteristics for several popular techniques, including learning from task properties, learning from model evaluations, and learning from prior models. Botvinick et al. (2019) summarized the implementation of deep meta-RL and proposed the future directions like combining episodic deep RL with meta-RL.

Compared to these surveys, we systematically and comprehensively review different deep RL algorithms rather than focusing on one specific branch, to show the relationship and development trend. Further, we systematically categorize the advanced RL methods and their applications, and propose promising directions to address the challenges of RL.

To summarize, this paper presents an extensive survey of RL with the following contributions:

1. We provide a detailed review over state-of-the-art RL methods and divide them into three categories: model-free, model-based, and advanced methods.

2. We systematically summarize the development of advanced RL and divide the algorithms into three categories including exploration, inverse RL, and transfer RL.

3. We detailedly analyze and discuss the application of RL, such as robotics, natural language processing, and computer systems.

4. We propose several open problems (e.g., data hunger issue and AGI issue) for future research as well as the analysis of each problem and the directions for further study.

2 Model-free reinforcement learning

The development of deep RL is still in its infancy. Academic research focuses on the deterministic and static environment where the states are mainly discrete and fully observed. Therefore, most RL jobs are based on model-free approaches. Model-free RL can estimate the state, value function, and reward function of the agent through a large number of samples to optimize the action policy $\pi_\theta(a|s)$ that aims to obtain more rewards by doing action a in state s. Because of its simple implementation and rich open resources, model-free RL has attracted more and more scholars to conduct further research.

In this section, we simply divide model-free RL into two scenarios: (1) RL based on the value function; (2) RL based on policy gradient.

2.1 RL based on the value function

2.1.1 Deep Q-network

Deep Q-network (DQN) (Mnih et al., 2013, 2015) is a typical representative of deep RL, which uses the convolutional neural network (CNN) (Krizhevsky et al., 2012) as a model and is trained with a variant of Q-learning (Kröse, 1995). DQN uses the maximum Q value as the low-dimensional action outputs to solve the confusing representation of high-dimensional state inputs such as raw pixels like the picture of the game.

Moreover, DQN reduces the bonus value and error term to a limited interval, which mitigates the instability of the value functions represented by a nonlinear network. Different from the Q-learning algorithm, DQN synchronizes the learning process and the training process, and the main improvements are summarized as follows: (1) experience replay buffer (Lin, 1992) is used to reduce the association between samples; (2) deep neural networks—target network—is used for action-value function approximation.

Experimental results show that DQN surpasses previous algorithms on most of the Atari 2600 games (Bellemare et al., 2013) and perform comparably to a human professional tester. When solving various types of visual perception based deep RL tasks, DQN uses the same set of network models, parameter settings, and training algorithms, which demonstrates that the method is highly adaptable and versatile (Gu SX et al., 2016).

2.1.2 Developments of DQN

A large number of improved algorithms have been proposed since the success of DQN. In this subsection, we focus mainly on representative methods related to the overall structure of the system, the construction of training samples, and the structure of neural networks.

Double deep Q-network (DDQN) (van Hasselt et al., 2016) reduces the risk of overestimation bias of Q-learning by decoupling selection and evaluation of the bootstrap action. Since experience transitions

are uniformly sampled from replay memory, DQN obviously fails to fully consider the importance of each sample. An improvement on the experience replay mechanism, DDQN (Schaul et al., 2016), tackles the problem by calculating the priority of each sample in the experience pool and increasing the probability of valuable training samples.

Limited by conventional architectures (e.g., convolutional networks, long short time memories (LSTMs), or auto-encoders), the DQN algorithm exhibits some other shortcomings, such as the lack of long-term memory capabilities. Hausknecht and Stone (2017) investigated the effects of adding recurrency to DQN by replacing the first post-convolutional fully connected layer with a recurrent LSTM. Wang ZY et al. (2016) expanded DQN based on a dueling architecture, which helps generalize across actions by separately representing state values and action advantages.

The multi-step bootstrap targets (Sutton, 1988; Sutton and Barto, 2018) used in asynchronous advantage actor-critic (A3C) (Mnih et al., 2016) shift the bias-variance tradeoff and help propagate newly observed rewards faster to earlier visited states. Noisy DQN (Fortunato et al., 2019) uses stochastic network layers for exploration. Distributional Q-learning (Bellemare et al., 2017) learns a categorical distribution of discounted returns instead of estimating the mean.

Since the aforementioned independent improvements of DQN are based on a shared framework, Rainbow (Hessel et al., 2018) could combine them plausibly. Experimental results show that the combination provides state-of-the-art performance on the Atari 2600 benchmark in terms of data efficiency and final performance.

From research on Rainbow, identifying priorities is the most important for the agent's performance. However, Schaul et al. (2016) used only a single actor to collect data, which is inefficient. Ape-X (Horgan et al., 2018) leverages a distributed architecture for deep RL at scale, and uses hundreds of actors to collect data and obtain a large number of replay buffers with different priority levels through various actors. The experiment shows that the performance of Ape-X is almost doubled in a shorter training time than Rainbow. Besides, it reveals that distributed computing is becoming one of the most important parts in deep RL.

2.2 RL based on policy gradient

REINFORCE (Williams, 1992) is the prototype of policy gradient (PG) algorithms. Compared with value-based RL, policy-based RL not only avoids the policy degradation caused by the value function error, but also is easier to apply in the continuous action space problem. Specifically, value-based methods, such as Q-learning and SARSA, require a one-step operation to calculate the maximum value, which can hardly be found in the continuous space or high-dimensional space. In addition, value-based methods learn implicit policies but policy-based RL methods can learn stochastic policies. That is, in the value-based method, the policies obtained through policy improvement are all deterministic policies, and will encounter some problems that cannot be resolved in some tasks like Rock-Paper-Scissors. Policy-based methods also have some common shortcomings: (1) data efficiency or sample utilization is low; (2) the variance is large, which makes it difficult to converge.

We introduce the typical improvements from two aspects: (1) improved framework based on actor-critic; (2) improved method based on the trust region.

2.2.1 Improved frameworks based on actor-critic

A classical and effective method to overcome the common drawbacks of policy-based methods is the actor-critic algorithm (Sutton and Barto, 2018), which learns a policy and a state-value function. The state-value function is used for bootstrapping, i.e., updating a state from subsequent estimates to reduce variance and accelerate learning. However, it has been a long-standing hurdle of RL to stably apply the simple but efficient design of actor critic methods to both continuous and discrete action spaces. By extending DQN and DPG (Silver et al., 2014), a deep deterministic policy gradient (DDPG) algorithm (Lillicrap et al., 2016) can learn competitive policies for tasks using low-dimensional observations (e.g., Cartesian coordinates and joint angles) that are based on the same hyper parameters and network structure. Moreover, the twin delayed DDPG algorithm (Fujimoto et al., 2018) is proposed as a deterministic algorithm which can concurrently work and has substantial improvement over DDPG.

However, DDPG, whose policy is deterministic, is particularly unsuitable in complex environments with noise interference because the policy is required generally to perform with a certain randomness. Another problem is that many commonly used model-free RL algorithms such as trust region policy optimization (TRPO) (Schulman et al., 2015), proximal policy optimization (PPO) (Schulman et al., 2017), or A3C, demand a number of new samples for each gradient step. Suffering from this strict requirement, it is extravagantly expensive to learn an effective policy and will be much worse with the increase of complexity of tasks. Soft actor-critic (SAC) (Haarnoja et al., 2018) is an effective approach for addressing these problems. SAC combines off-policy updates with a stable stochastic AC formulation and uses a maximum entropy framework to augment the standard maximum reward RL objective. A substantial improvement can be attained in performance and sample efficiency over off-policy and on-policy prior methods, because there are several special advantages of the RL algorithm based on maximum entropy:

1. An initialization for more complex specific tasks. The policy learned by maximum entropy can be used as an initialization for more complex specific tasks and to learn a way to solve tasks.

2. Stronger exploration capabilities. It is obvious that maximum entropy makes it easier to find better patterns under a multimodal reward.

3. High robustness and stronger generalization. Maximum entropy requires exploring various optimal possibilities from different ways and makes it easier to do adjustments in the face of interference.

2.2.2 Improved methods based on the trust region

The policy gradient method has the problem that the policy is difficult to update stably in the case of unstable data because of using a neural network as the nonlinear function approximator. In recent years, the academic community has introduced the trust region method into RL and enabled substantial performance improvements in various experimental scenarios.

Based on the conclusion of conservative policy iteration (Kakade and Langford, 2002), TRPO computes the maximum value of total variation divergence as the learning rate and considers the relationship between total variation divergence and Kullback-Leibler divergence, so as to extend the mix policy into a general stochastic strategy. The generalized advantage estimator (Schulman et al., 2016) uses two factors as tunable parameters in the discounted Markov decision process (MDP) further to implement the trade-off between bias and variance in the actor-critic algorithm.

One trouble for TRPO is the large amount of interaction with the environment. Wang ZY et al. (2017) first addressed the problem from sample inefficiency at scale by introducing an actor-critic with experience replay (ACER). Unlike TRPO, which restricts policy updates by constraints, ACER maintains a sliding average that represents the past policy and keeps policy updates from deviation from this mean. Similarly, actor-critic using the Kronecker-factored trust region (Wu et al., 2017) optimizes both the actor and the critic using Kronecker-factored approximate curvature with the trust region, while constrained policy optimization (Achiam et al., 2017) uses a constrained MDP.

TRPO fails on complex calculations or compatibility with some architectures (e.g., noise or parameter sharing). It is much simpler to implement PPO, since it uses an objective function with a clipped probability ratio to form a pessimistic estimate based on first-order optimization, which, however, may result in sample inefficiency. Distributed PPO (Heess et al., 2017) is presented further to explore how a rich environment can help promote the learning of complex behaviors. As in A3C, data collection and gradient calculation are deployed to multiple distributed workers. This not only improves performance and scalability but also enables stable behaviors in a rich and varied environment.

Combining the advantages of on-policy and off-policy, path consistency learning (PCL) (Nachum et al., 2017a) enhances exploration capabilities based on a relationship between softmax temporal value consistency and policy optimality under entropy regularization. Improved on PCL, Trust-PCL (Nachum et al., 2017b) adds the discounted relative-entropy trust region into constrained optimization. This not only ensures the stability of optimization but also makes full use of off-policy data to improve sample efficiency.

3 Model-based reinforcement learning

It is a fact that knowing the transition dynamics $p(s_{t+1}|s_t, a_t)$ makes things easier. The dynamics are called models. Model-based methods are the algorithms that learn the transition dynamics that decide which next state s_{t+1} would be after doing action a_t in the current state s_t. Then methods will figure out how to choose actions. In brief, this kind of algorithm learns models of system dynamics and uses optimal control to choose actions. Model-based RL develops from the field of optimal control. Usually, specific problems are built by models such as the Gaussian process and Bayesian network, and then are solved through machine learning methods or optimal control methods, e.g., model predictive control (MPC), linear quadratic regulator (LQR), and linear-quadratic-Gaussian control.

Compared with model-free RL, model-based RL learns a value function or policy in a data-efficient way and does not need continuous interaction with the environment. However, it may suffer from the issue of model identification and lead to an inaccurate description of the real environment. In this section, we divide the context of model-based RL into three scenarios and systematically analyze their advantages and disadvantages.

Table 1 shows different algorithms that require different numbers of samples and different computation time. It is a complicated question. This table is a rough guide but we think it can be a starting point. We list the data of these algorithms based on the standard benchmark task "half-cheetah" including the numbers of steps and episodes and total time for learning. Fig. 1 reveals the different usage scenarios of model-based and model-free methods. If we have a simulator and the simulation cost is negligible compared to the training cost (it is much cheaper to take one simulation step than one gradient step on the model), it is a really good choice to go with online-policy or policy-gradient

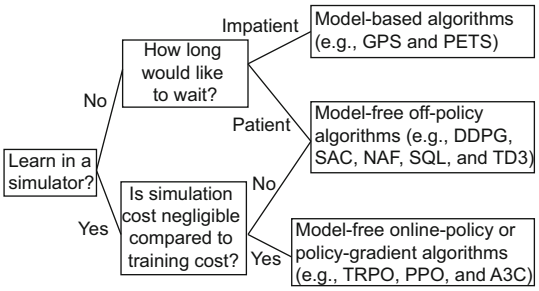

Fig. 1 Different usage scenarios of model-based and model-free methods

algorithms, such as TRPO, PPO, and A3C, because these algorithms are easier to tune and have fewer hyper parameters and then converge more reliably. If we have a simulator but the simulation is expensive, then we might opt for model-free off-policy algorithms like DDPG, NAF (Gu SX et al., 2016), and SQL (Haarnoja et al., 2017). There are many different choices for all policy-model-free algorithms that can be Q-learning-based or extra-critic-based. Finally, if we do not have a simulator, the main question is how long an acceptable waiting period is. If the answer is not long (impatience), the model-based algorithms like guided policy search (GPS) (Levine and Koltun, 2013) and probabilistic ensembles with trajectory sampling (PETS) (Chua et al., 2018) are good choices. Otherwise, you can choose the model-free off-policy algorithms that require somewhat fewer assumptions and may be a little bit less domain-specific.

3.1 Global and local models

For model-based algorithms, the first question is which one should be fitted if the dynamics is unknown: global dynamics models or local dynamics models? In this subsection, we introduce the related algorithms and the comparison between these two models.

In spite of some advantages such as cheap computation at runtime, global model methods often fail on numerical stability, especially in

Table 1 Comparison of different reinforcement learning algorithms

	Method	Reference	Number of steps	Number of episodes	Time
Model-free	Fully online (e.g., A3C)	Wang ZY et al. (2016)	100 000 000	100 000	∼15 d
	Policy gradient (e.g., TROPO)	Schulman et al. (2017)	10 000 000	10 000	∼1.5 d
	Value estimation (e.g., DDPG and SAG)	Gu SX et al. (2016)	1 000 000	1000	∼3 h
Model-based	E.g., PETS and GPS	Chua et al. (2018)	30 000	30	∼5 min

stochastic domains, because they iteratively collect data using MPC and directly backpropagate them into policy. There has been some effort to overcome this shortcoming. GPS (Levine and Koltun, 2013) uses trajectory optimization to direct policy learning and avoid poor local optima. Levine et al. (2016) developed GPS and a CNN architecture to reduce real-world interaction and to decrease the number of training samples. Nagabandi et al. (2018) initialized a model-free learner with the help of special deep neural network dynamics models, which can combine sample efficiency of model-based approaches with the high task-specific performance of model-free methods. The results on MuJoCo locomotion tasks show that the algorithm has excellent sample efficiency, and can accelerate model-free learning on high-speed benchmark tasks.

In most of the state space, the planner in global models may seek out the region where the model is erroneously optimistic, thus requiring a very accurate model to converge on a good solution. Moreover, it is often much more difficult to obtain a proper model than learning a policy especially when the environment is hard to describe. Therefore, more and more attention is paid to local models with constraints. Local models need to figure out which controller is executed to obtain the right data and how to ensure the whole models not to diverge horribly. Here we list some typical algorithms.

A model-based iterative linear quadratic regulator (iLQR) (Levine et al., 2015) is extended on GPS and can learn a range of dynamic manipulation behaviors with highly general policy representations, without using known models or example demonstrations. Combining the advantages of model-free RL, Dyna-Q (Sutton and Barto, 2018) is a classical integrated architecture where a model is leveraged to update Q-values. Probabilistic inference for particle-based policy search further allows for almost arbitrary models and policies by simultaneously matching the performance of previous data-efficient learning algorithms.

To adopt good points from both model-free and model-based RL algorithms, Chebotar et al. (2017) focused on time-varying linear-Gaussian policies, and integrated a model-based LQR algorithm with a model-free path integral policy improvement algorithm. Imagination-augmented agents (I2As) (Racanière et al., 2017) learns to interpret environment models to augment model-free decisions. Compared to Monte Carlo tree search (MCTS) (Silver et al., 2016), I2As shows improvements in data efficiency and robustness by mitigating the problem of model misspecification.

3.2 Uncertainty-aware model

There is a performance gap between pure-model-based and model-free methods. Compared with model-free methods that require 10 days, model-based methods enable a complete training process using only 10 min in real time. However, model-free methods can achieve much better performance, differing by at most three orders of magnitude (Nagabandi et al., 2018). The main reason is overfitting. The uncertainty-aware model is an effective approach for solving this problem. Based on the representation of uncertainty, we divide it into two distinct classes, aleatoric uncertainty (inherent system stochasticity) and epistemic uncertainty (subjective uncertainty due to limited data). There are two main directions, estimating mode uncertainty and using output entropy to build uncertainty-aware models. We introduce the former in this section and the latter in Section 4.1.1.

By learning a probabilistic dynamic model and explicitly incorporating model uncertainty into long-term planning, probabilistic inference for learning control (PILCO) can cope with very little data and facilitate learning from scratch in only a few trials. Blundell et al. (2015) used variational Bayesian-learning to estimate the uncertainty in a nerual network (Fig. 2). Compared to feedforward neural networks, this method not only mitigates the issue of overfitting but also correctly assesses the uncertainty in the training data. Similarly, relying on developments in Bayesian deep learning, the approach in Gal et al. (2017) allows the agent to adapt

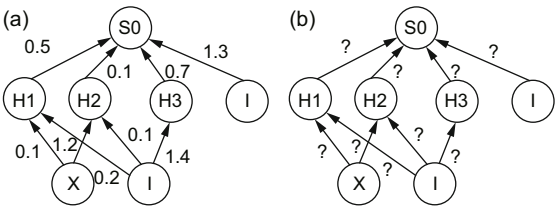

Fig. 2 Process to estimate the uncertainty in a neural network: (a) each weight has a fixed value provided by classical backpropagation; (b) each weight is assigned a distribution provided by Bayesian methods

its uncertainty dynamically and observes more data using a continuous relaxation of a dropout's discrete masks. PETS (Chua et al., 2018) combines uncertainty-aware deep network dynamics models with sampling-based uncertainty propagation and finally brings these components together in a deep RL framework that reaches the asymptotic performance of the model-free RL methods on benchmark control tasks.

There is a growing interest in combining model-free and model-based approaches in RL to achieve algorithms with high performance but low sample complexity. Model-based value expansion (Feinberg et al., 2018) allows imagination to fixed depth to control the uncertainty of the model. Stochastic ensemble value expansion (Buckman et al., 2018) ensures that the model is used only when doing so does not introduce significant errors by dynamical interpolation between model rollouts of various horizon lengths.

3.3 Model for complex observations

Model-based RL has proven to be a data-efficient approach for learning control tasks, but it is difficult to use in partially observable MDPs with complex observations such as images. This is because agents have to make a decision based on the observation rather than the accurate state of the environment.

Spatial autoencoder architectures (Lange et al., 2012; Finn et al., 2016b) are presented to learn in the latent space and autonomously learn low-dimensional embedding of the image. However, there is an inevitable issue that the autoencoder might not recover the right representation and is not suitable for model-based RL. To tackle this difficult problem, embed to control (E2C) (Watter et al., 2015) applies variational autoencoders with iLQR in a latent space and turns the problem of locally optimal control in high-dimensional nonlinear systems into a low-dimensional latent state space.

The method that combines deep action-conditioned video prediction models with model-predictive control (Finn and Levine, 2017) is the first instance of robotic manipulation to learn directly in the observation space. It trains the agent with entirely unlabeled data and plans for actions that move user-specified objects in the environment to user-defined locations, both of which could help generalize to new, previously unseen objects. A video prediction model (Ebert et al., 2017) can also keep track of objects by incorporating temporal skip connections and achieve significant advance in the range and complexity of skills entirely with self-supervised robot learning. Since accurate forward prediction could be very expensive (Chua et al., 2018; Nagabandi et al., 2018), SOLAP (Zhang et al., 2019) uses simple models, typically linear models, to provide gradient directions for local policy improvement rather than forward prediction and planning.

4 Advanced reinforcement learning

4.1 Exploration

In many complex RL tasks, agents face the challenge of balancing exploration and exploitation when interacting with unknown dynamics. With the rapid development of RL, various effective and scalable approaches have been proposed to overcome the drawback of lacking exploration. We will introduce the typical algorithms of each category and analyze their advantages and disadvantages.

4.1.1 Optimistic exploration

Since unexplored actions could bring a better reward, it would be valuable to optimize them to effectively increase exploration. Therefore, how to quantify the state novelty of these never-accessed states is key to solving such complex problems.

Frequency-based distribution reward mechanisms have been widely applied. These count the frequency of occurrences of an action (state-action) as a bonus:

$$r^+(s,a) = r(s,a) + B(N(s)), \tag{1}$$

where $N(s)$ represents the frequency of occurrences of state s. $B(N(s))$ represents a bonus that decreases with $N(s)$. Optimistic exploration uses $r^+(s,a)$ instead of $r(s,a)$ as reward with any model-free algorithm. Different bonuses have the same essential feature of tending to choose the optimal or the latest action. Apart from the upper confidence bound, used by AlphaGo as a bonus in MCTS, there are some other bonuses such as the model-based interval estimation with exploration bonus (Strehl and Littman, 2008) and Bayesian exploration bonus (Kolter and Ng, 2009).

The optimistic initial value method (Sutton and Barto, 2018) achieves exploration by increasing the initial value of the value function instead of counting the frequency of the state. The gradient bandit algorithm (Sutton and Barto, 2018) uses the entropy value rather than the value function to represent uncertainty. The algorithm adjusts the entropy of each action through the rewards obtained, and tends to select an action with a larger entropy value.

In some complex high-dimensional continuous environments (e.g., StarCraft II), it is almost impossible for the exact same state to appear twice. This will cause the above approaches to be meaningless because almost every state counts one. Derived from a density model, the notion of pseudo-count was introduced to generalize count-based exploration bonuses for non-tabular RL agents (Bellemare et al., 2016). Since then, a large amount of effort has been put into improving the pseudo-count algorithm. Ostrovski et al. (2017) improved the work by the count-based exploration with neural density models. Hash-based count (Tang et al., 2017) maps states to hash codes and count states' occurrences with a hash table. By exemplar model exploration (Fu et al., 2017a), classifiers are trained to discriminate each visited state against all others and the new state is implicitly compared to the old state.

4.1.2 Posterior sampling exploration

Exploring with random actions (e.g., epsilon-greedy) suffers from oscillating back and forth. It might not go to a coherent or interesting place, while exploring with random Q-functions can commit to a randomized but internally consistent strategy for an entire episode. Inspired by this, posterior sampling algorithms are proposed for more targeted exploration.

The Thompson sampling approach (Chapelle and Li, 2011) samples the Q-value of each action from the prior distribution. Then, the reward probability of each action is considered as a Beta distribution, which helps provide guidance on tweaking the posterior and achieve a smaller regret. Bootstrapped DQN (Osband et al., 2016) combines deep exploration with deep neural networks to overcome the shortcoming of incompatibility with nonlinearly parameterized value functions. Fig. 3 shows the architecture of bootstrapped DQN.

Bootstrapped DQN trains bootstrap data on multiple shunt networks to randomize different value functions. This way of training contributes to a balance between learning from noise and exploring complex state/action spaces. Through this distributed deep exploration, bootstrapped DQN can fully guarantee the exploration of various strategies, produce diverse samples, and better generalize to the unknown state space.

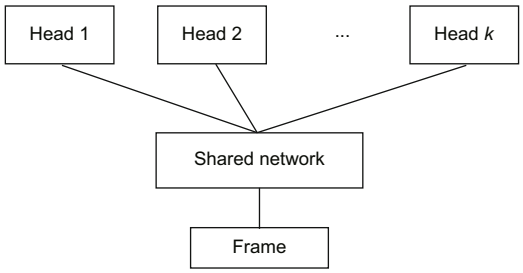

Fig. 3 Architecture of bootstrapped DQN
Head represents Q-function. The shared network learns a joint feature representation from all data. Frame is a replay buffer storing samples

4.1.3 Information gain exploration

In some complex environments, sampling one action can help the agent assess other actions. Based on this, for information gain exploration, the advantage of information structures is fully considered, which helps the agent learn more efficiently in difficult tasks.

Information-directed sampling (Russo and Roy, 2014) measures the mutual information between the true optimal action and the next observation, and then selects an action to quantify the amount learned. Each selected action is sampled in a manner that the ratio between expected single-period regret and the measure of information gain is minimized, which will also balance exploration and exploitation to some extent.

However, it is impossible to construct a shaped reward function for the reason that the rewards of the environment are extremely sparse or missing altogether in many real-world scenarios. Compared to the aforementioned exploration approaches, variational information maximizing exploration (VIME) (Houthooft et al., 2017) approximates the probability of the dynamic model and greatly reduces the reliance on rewards using variational inference in Bayesian neural networks. The inner exploration mechanism where the reward function is modified by the information gain motivates the agent to explore

the unknown domain and enables integration with other methods such as TRPO.

Similar to the idea of VIME, curiosity-driven learning (Pathak et al., 2017; Burda et al., 2019) generates an intrinsic reward signal based on how hard it is for the agent to predict the consequences of its own actions (i.e., predict the next state given the current state and the executed action). The prediction error of the forward dynamics model is used as an intrinsic reward to encourage the agent's curiosity. The experiments reveal that curiosity-driven learning could achieve significant performance across a variety of continuous control tasks even in the environment with very sparse rewards.

4.2 Inverse RL

Usually we learn a transition model assuming that the reward model is already known. However, there will be a huge impact on performance once the reward function is not designed properly. Inverse RL (IRL) (Ng and Russell, 2000) is introduced to learn a proper reward function from the observed expert examples. However, there are some challenges in IRL: (1) The problem is under-defined and lacks prior knowledge; (2) It is difficult to evaluate a learned reward; (3) Demonstrations may not be precisely optimal. In the following, we discuss the solutions based on maximum margin and maximum entropy.

4.2.1 IRL based on maximum margin

Apprenticeship learning (Abbeel and Ng, 2004) uses the maximum marginal method to find the current reward function from an expert example. There is a guarantee that the optimal policy obtained under the reward function is near the expert example policy.

Maximum margin planning (MMP) (Ratliff et al., 2006) attempts to automate the mapping from perception features to costs by structured maximum-margin classification. In spite of some improvements with tricks (e.g., support vector machines), MMP still has troubles in calculation when iteratively solving MDP. To handle this issue, IRL through structured classification (Klein et al., 2012) constrains each action at each state instead of constraining the solution of an MDP and uses the so-called feature expectation of the expert as the parameterization of the score function of a multiclass classifier.

Further, neural inverse RL (NIRL) (Xia and El Kamel, 2016) focuses on IRL with large-scale high-dimensional state spaces. With the help of a neural network, NIRL can not only generalize the expert's behaviors into unvisited regions of the state space but also express an explicit policy representation easily even for the stochastic expert policy.

In all, methods based on maximum margins tend to be ambiguous. For instance, many different reward functions lead to the same expert policy. In this case, the reward function learned often has a random preference. The challenges are summarized as follows: (1) Maximizing the margin is a bit arbitrary; (2) There is no clear model of expert suboptimality; (3) Deep planning is not applicable for the messy constrained optimization problem.

4.2.2 IRL based on maximum entropy

The method of maximum entropy can avoid ambiguity problems because its probability is distributed without any assumption about the distribution of any other location information except constraints. It may cause ambiguity when choosing a distribution over decisions under the constraint of matching the reward value of the demonstrated behavior. Maximum entropy IRL (MaxEnt IRL) (Ziebart et al., 2008) is proposed to solve the problem by employing the principle of maximum entropy.

Although MaxEnt IRL solves the ambiguity problem, it does not adapt to large and continuous states and action spaces or meet the requirements for effective learning under unknown dynamics. Methods based on maximum entropy are also difficult to apply in practical applications because: (1) The learning of the reward function requires artificially selected features (the choice of features is very difficult for many practical problems); (2) Many subcycles of IRL contain forward RL, which is a difficult problem. To address the former challenge, Ziebart et al. (2008) exploited the representational capacity of neural networks to approximate complex and nonlinear reward functions. To address the latter challenge, guided cost learning (Finn et al., 2016a) further formulates an efficient sample-based approximation instead of forward RL.

Generative adversarial imitation learning (GAIL) (Ho and Ermon, 2016) maps the reward function target in IRL in the context of the

generative adversarial network (GAN). The strategy model of GAIL acts as a production model in GAN, generating actions with state as the input. The reward function model of GAIL can serve as a discriminant model for discriminating the extent to which actions approximate expert actions. The experimental results show that GAIL could achieve significant performance in imitating complex behaviors in large, high-dimensional environments. Adversarial inverse reinforcement learning (AIRL) (Fu et al., 2017b) provides simultaneous learning of the value function and reward function. In contrast to GAIL, AIRL makes adequate use of the efficient adversarial formulation and can also recover a generalizable and portable reward function.

4.3 Transfer RL

A fundamental problem in artificial intelligence is that it cannot learn as efficiently as a human. Many RL algorithms demonstrate superhuman performance but require millions of training samples. Many transfer RL algorithms are presented motivated by an intuition that useful knowledge might be acquired based on prior tasks to solve a new task. This kind of algorithm aims to rapidly learn an optimal policy in a new environment using only a small amount of available data.

We divide transfer RL into three categories according to the choice of the source domain: (1) forward transfer that trains on one task and transfers to a new task; (2) multi-task transfer that trains on many tasks and transfers to a new task; (3) meta-RL that learns to learn from many tasks.

4.3.1 Forward transfer

The simplest method of forward transfer is just to try with the best hope (Levine et al., 2016). Policies trained for one set of circumstances might just work and could deal with new tasks successfully with good luck because sometimes there is enough variability during training to generalize.

Finetuning remains one of the most popular choices for transfer learning with neural networks. First, a model is pretrained on a source domain where data is often abundant. Second, the output layers of the model are adapted to the target domain. Finally, the network is finetuned via backpropagation. For example, in an approach of pre-training for diversity (Haarnoja et al., 2017), it learns all the possible ways of a given task to achieve a particular policy. This policy can serve as a good initialization for finetuning to a more specific behavior (i.e., first learn all the ways that a robot could move forward, and then use this as an initialization to learn separate running and bounding skills).

Unfortunately, the finetuning approach often results in overfitting when finetuning with a little bit of experience in large deep networks. Progressive neural networks (Rusu et al., 2016b) mitigate overfitting and solve expressivity problems by construction. This framework retains a pool of pre-trained models throughout the training process and learns lateral connections from these to extract useful features for the new task. Fig. 4 shows the architecture of progressive neural networks. The gray box labeled "a" indicates the adapters, which keep the hidden layer activation values of the front row consistent with the original input dimensions.

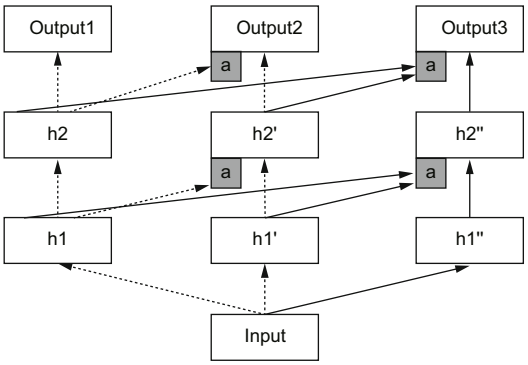

Fig. 4 A simple progressive neural network

It is a fact that agents will transfer better if they see more diversity during the training. To achieve better performance, the methods make the network unable to distinguish observations from the source domain and target domain by modifying the source domain and using a modest amount of target domain data for transfer. The ensemble policy optimization algorithm (Rajeswaran et al., 2017) can adapt the probability over source domains in the ensemble using the data from the target domain and randomizing physical parameters of simulated source domains. The approach can help learn more robust policies and generalize to a broad range of possible target domains. For perception-based tasks, the methods that prepare for the unknown (Yu WH et al., 2017; Peng et al., 2018b) use recurrent neural networks

(RNN) to output unknown parameters in the true environment (such as dictionary/mass) to precompute many possible situations that the robot might encounter when acting in the real world to reduce the interaction with the real world.

4.3.2 Multi-task transfer

Typical applications of RL focus more on mastery than on one-shot learning and require a substantial number of training episodes. Multi-task transfer provides a way to address these challenges and is closer to what people do—build a lifetime of experience.

One of the simplest solutions is to learn a model that can simultaneously perform many tasks. Online dynamics adaptation (Fu et al., 2016) combines prior knowledge from previous tasks with online adaptation of the dynamics model. This model-based approach not only decreases the demand for training data using the model, but also performs one-shot learning of new tasks using the robot's experience on other tasks, without requiring explicit domain knowledge or demonstrations be provided by the designer.

However, sometimes learning a model can be very difficult. Model-free approaches have been proposed to combine the policies after training each MDP separately (Parisotto et al., 2016; Rusu et al., 2016a). This kind of algorithm exploits DQN and model compression techniques to train a single policy network and learns how to act in a set of distinct tasks using the guidance of several expert teachers.

Architectures with reusable components can be designed to make full use of the characteristics of tasks if they have shared parts and distinct parts. Since the above kinds of methods build a single network for all tasks, modular policies (Devin et al., 2017) can achieve better performance by decomposing neural network policies into task- and robot-specific modules. The algorithm trains the same task-specific component across all robots and the same robot-specific component across all tasks. Then the robot- and task-specific modules will be mixed and matched to execute new tasks. Fig. 5 shows the process of modular policies. This special architecture allows the agent to share task information (such as perception) between robots and share robot information (such as dynamics and kinematics) between tasks.

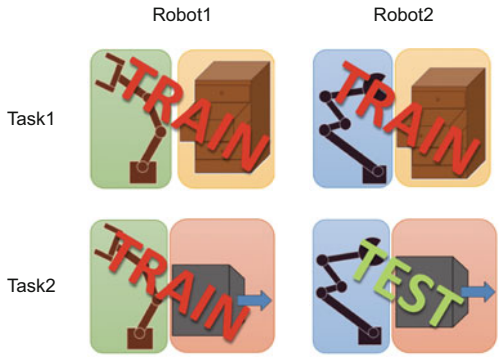

Fig. 5 Process of modular policies
After training robot1 with task1 and task2 and training robot2 with task1, robot2 can execute new task2 when testing by mixing the robot- and task-specific components

4.3.3 Meta-RL

Meta-RL offers a feasible framework to address few-shot learning in a very complicated environment that demands strategic and tactical thinking. Let $M_i = (S, A, T_i, R_i)$ represent an MDP with state space S, action space A, transition probability distribution T_i, and reward function R_i. In meta-RL, we consider a family of MDPs $M = \{M_i\}_{i=1}^{N}$, which comprise a task distribution $\tau_i \sim p(\mathcal{T})$ such that each MDP shares the statistical regularity of $p(\mathcal{T})$. Meta-RL aims to improve the learning efficiency for novel subsequent tasks by learning a meta-item from the family of MDPs such as policy π_θ, model T_θ, and the reward function. We introduce two kinds of most representative meta-RL algorithms and then list some main advanced improvements.

Almost at the same time, Duan et al. (2017) and Wang JX et al. (2017) defined a new concept, deep meta-RL, and put forward recurrent models called RL^2, which are encoded in the weights of the RNN and learned slowly through a general-purpose RL algorithm. The main idea of RL^2 is training neural networks with all historical trajectories (state, action, and reward) and automatically determining the information at the task level (meta level). This design can significantly speed up the training process of the new task and is thought as one of the most basic changes from RL to meta-RL. Fig. 6 illustrates the process of interaction between an agent and the environment. Further, a simple neural attentive learner (SNAIL) (Mishra et al., 2018) is proposed to solve the problem of information loss in the above original models. To aggregate information from experience and pinpoint specific pieces of

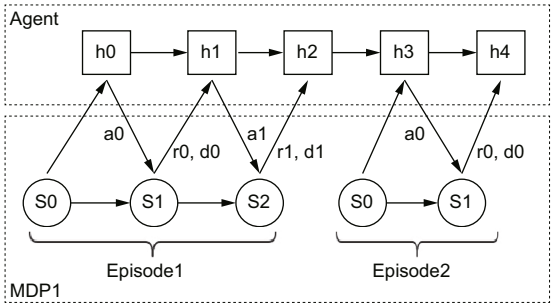

Fig. 6 Process of interaction between an agent and the environment
For each episode in each MDP, a fresh S0 is drawn from the initial state distribution specific to the corresponding MDP. Agent is trained with historical trajectories. At the end of an episode, the hidden state of the policy is preserved to the next episode, but not preserved between trials

information, SNAIL adjusts the architecture of RNN with LSTM and uses a novel combination of interleaved one-dimensional convolutions and soft attention. E-RL^2 (Stadie et al., 2018) adds an indicator to the standard sum of discounted returns, returning one if the episode helps account for the impact of this initial sampling distribution and 0 otherwise. Experiments show that this encourages the RNN to account for the impact of casting a wider sampling distribution on the final meta-reward. ReBAL (Nagabandi et al., 2019) uses a recurrent model to learn the transition model instead of the action policy to perform well on the simulated continuous control tasks.

Recurrent models have a variety of choices of design in the architecture and are usually general and expressive. However, they often fail in dealing with complex tasks and demand complex models and impractical data. Another typical meta-RL approach is model-agnostic meta-learning (MAML) (Finn et al., 2017a). To overcome the drawbacks of recurrent models, MAML uses a fixed optimizer and learns a set of base parameters that can be adapted to minimize any task loss by a few steps of gradient descent.

MAML can be directly applied to any learning problem and model that is trained with a gradient descent procedure because of the fixed optimizer. Since MAML is simple but powerful, it attracts great interest among researchers. Improving exploration is an attractive perspective. E-MAML (Stadie et al., 2018) highlights the importance of correctly differentiating through sampling distributions in meta-RL. Model agnostic exploration with structured noise (MAESN) (Gupta et al., 2018) injects structured stochasticity into MAML to acquire a latent exploration space. Different from MAESN that adds extra parameters as the embedding space, fast context adaptation via meta-learning (CAVIA) (Zintgraf et al., 2019) uses the same network to learn the embedding during a backward pass and make predictions during a forward pass. Combined with inverse RL, SMILe (Ghasemipour et al., 2019) is the first efficient method for meta-IRL that scales to the function approximator setting. To tackle the problem of efficient off-policy meta-RL, there are also some novel algorithms. GrBAl (Nagabandi et al., 2019) combines MAML with model-based algorithms (e.g., MPC) to learn the transition model. Probabilistic embeddings for actor-critic RL (PEARL) (Rakelly et al., 2019) integrates online inference of probabilistic context variables with existing off-policy RL algorithms.

Apart from the improvement based on the framework of MAML, there are some other novel approaches. The meta-MDP approach to exploration (Garcia and Thomas, 2019) models the problem of searching for an optimal exploration policy as an MDP, and separates the policies into an exploration policy and an exploitation policy to achieve lifelong learning.

In all, deep meta-RL is a promising research direction and is being researched using a variety of meta-angles and numerous exploration methods to achieve better performance of both meta-training and adaptation efficiency. Fig. 7 provides a representation of relationships between improved deep meta-RL algorithms.

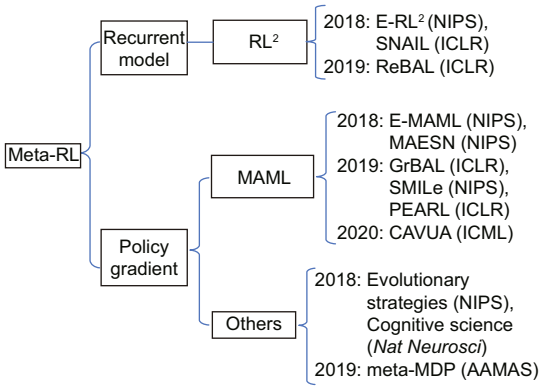

Fig. 7 Relationship between deep meta-RL algorithms

5 Applications

In the early stages of RL development, DQN was used mainly in various two-dimensional video games such as Atari games (Mnih et al., 2016; Schaul et al., 2016; Hessel et al., 2018; Horgan et al., 2018). With AlphaGo successfully defeating the world champion of chess using deep neural networks and MCTS, there has been increasing interest in RL to improve the performance in dealing with complex tasks in different fields in recent years.

In the field of games, StarCraft II (Vinyals et al., 2017) offers a challenging multi-agent environment with multiple players interacting to explore RL algorithms and architectures. Deep meta-RL can also be leveraged in games to do few-shot learning, such as by imitating human Atari game play from a single recorded action sequence (Pohlen et al., 2018) or online videos (Aytar et al., 2018).

Deep RL has achieved significant success in various fields. We outline current representative RL applications including robotics, natural language processing (NLP), and computer systems in the following.

5.1 Robotics

Robotics is a classic area for RL. RL can implement behavioral control of complex robots in the simulation environment, thus enabling realistic responses to perturbations and environmental variation. Apart from Atari and simple agents in Mujoco (e.g., half-cheetah, ant, and spider), DeepMimic (Peng et al., 2018a) further develops challenging multiskilled agents including multiple characters (e.g., human, Atlas robot, bipedal dinosaur, and dragon) and a large variety of skills (e.g., locomotion, acrobatics, and martial arts).

In addition, RL has obtained numerous research results in robot control tasks in real situations. A range of real-world tasks are contact-rich and require close coordination between vision and control, such as stacking tight-fitting Lego blocks and screwing bottle caps onto a bottle. The improvements (Levine et al., 2015) control manipulation to complete these tasks by reducing the sample count and automating parameter selection in GPS. An RNN with LSTM (Rahmatizadeh et al., 2016) helps the controller learn from virtual demonstrations and successfully performs the manipulation tasks on a physical robot. By closed-loop vision-based control (Kalashnikov et al., 2018), re-grasping strategies are automatically learned, probing an object and repositioning objects to find the most effective grasps and perform other non-prehensile pre-grasp manipulations.

However, training data of real robots is scarce for real scenarios. The method which combines knowledge from previous tasks with online adaptation of the dynamics model (Fu et al., 2016) helps solve a variety of complex robotic manipulation tasks in a single attempt. Multiple robots (Gu SX et al., 2017a; Yahya et al., 2017) learn collaboratively to sample and train in parallel. Manipulating the source domain (Peng et al., 2018b) narrows the gap between simulation and real physical systems. Al-Nima et al. (2019) produced suitable road tracking actions based on RL by collecting input states from forward car facing views.

Based on transfer learning (Devin et al., 2017), robots can share task-specific modules across robots and robot-specific modules across all tasks. The improved methods of meta-RL (Finn et al., 2017b; Yu TH et al., 2018) enable the agents to learn rapidly from little data in new environments. Learning to adapt to dynamic real-world environments (Nagabandi et al., 2019) further alleviates the problem of missing training data and has better generalization ability in dealing with robotic manipulation tasks.

5.2 Natural language processing

RL methods have broad application prospects in the domain of NLP and have been successfully applied in the fields of neural machine translation (NMT), dialog systems, and speech generation.

NMT systems generally rely on aligned parallel training corpora, but such parallel data is costly to collect in practice. This contradiction may result in heavy limitation in scale and constrain related research and applications. Dual learning for machine translation (He et al., 2016) tackles the data hunger issue in NMT by training translation models from unlabeled data through RL (e.g., PG). Using this mechanism, the monolingual data can play a similar role to the parallel bilingual data, and significantly reduces the requirement on parallel bilingual data during the training process.

Benefitting from the development of transfer learning and deep meta-RL, universal NMT (Gu JT et al., 2018a) uses a transfer-learning approach

to share lexical and sentence representations across multiple source languages into one target language. This enables the low-resource language to use the lexical and sentence representations of the higher resource languages. Further, Gu JT et al. (2018b) first extended a deep meta-RL algorithm (e.g., MAML) into low-resource NMT. The model can learn to adapt to low-resource languages based on multilingual high-resource language tasks. The results show that the method significantly outperforms the prior multilingual approaches and enables the training of a competitive NMT system with only a fraction of training examples.

The task of chatbots in dialogue systems is to mimic human-human interactions with extended conversations (Shum et al., 2018). A modified version of the episodic REINFORCE algorithm (Dhingra et al., 2017) explores and learns the policy and the posterior probability over the knowledge base entries for correct retrievals to select dialogue acts. Moreover, although many text-to-speech (TTS) systems (Ping et al., 2018; Skerry-Ryan et al., 2018) have produced high-quality samples for speakers present in the training set, generalizing to new speakers given only a few seconds of audio remains a challenge. An adaptive TTS approach (Chen YT et al., 2019) is presented based on MAML to highly restore the speaker's voice in new scenes using very few speech samples. Similarly, the DeepVoice model (Ping et al., 2018) is improved by predicting the embedding with an encoding network and fitting the embedding based on a small amount of adaptation data (Arik et al., 2018).

5.3 Computer systems

Computer systems present many challenging problems for RL, including time-varying state or action spaces (e.g., dynamically varying number of jobs and machines in a computer cluster), structured data sources (e.g., graphs to represent data flow of jobs or a network's topology), and highly stochastic environments (e.g., random time-varying workloads). Here, we summarize some typical RL methods used in computer systems and show that RL could provide significant real-world benefits in this domain.

Tackling multi-resource cluster scheduling with a PG algorithm (Mao et al., 2016) optimizes various objectives like average job slowdown or completion time in an online manner with dynamic job arrivals, and validates the approach via simulation. Chen L et al. (2018) proposed a two-level system called "automatic traffic optimization (AuTO)" to solve the scalability problem in data center traffic. One level is the peripheral system and the other is the central system. They leverage the long-tail distribution, which is a feature of data center traffic. The central system uses a DDPG algorithm to help the peripheral system choose the optimal parameters. This allows AuTO to solve the scalability problem of traffic optimization in data centers and achieve superior performance.

Motivated by prior applications, a scalable RL model with the graph embedding technique (Mao et al., 2019a) is trained by the PG algorithm to deal with the issue of continuous stochastic job arrivals. The model could help learn workload-specific scheduling algorithms without any human instruction beyond a high-level objective (e.g., minimizing average job completion time). Further, inspired by much potential for RL to improve the performance, an open extensible platform Park (Mao et al., 2019b) defines the MDP formulation (e.g., state, action space, and reward function). Park connects to a suite of real-world computer systems and lowers the barrier of entry for machine learning researchers to innovate based on deep RL in computer systems.

6 Challenges and the future

RL is one of the closest things that look anything like AGI. However, there are still numerous problems and many of them are fundamentally difficult:

1. Inefficient sample

As listed in Table 1, an extremely large number of training samples are required to allow the model to reach a certain level. A lot of time is taken for an Atari game that most humans pick up within a few minutes. Rainbow DQN (Section 2.1.2) takes long time to train the model and needs about 83 h of play experience to exceed the human level. Model-based RL is computationally expensive and has its own planning fallacy: learning a good policy usually needs more samples and the practical level of sample efficiency is usually much lower than the expected result. Even though deep meta-RL algorithms (Section 4.3.3) achieve few-shot learning on a new task, they need to train with a lot of data to find a good generalization model.

2. A demanding reward function

RL assumes the existence of a reward function, which must be defined exactly to ensure that the agent can do the right thing all the time. However, reward function design is often difficult: (1) A suitable prior is needed; (2) Perfect knowledge of all object states is required; (3) A good definition of the problem is necessary. Moreover, sometimes too much effort on the reward function may introduce new biases. Even given a good reward, it is hard to escape from local optima. The several mentioned intuitively pleasing ways of exploration (Section 4.1) mitigate the problem to some extent. However, as far as we know, none of them could work consistently across all environments. In addition, IRL and imitation learning do not need a reward function, but their performances are usually limited and unsatisfactory.

3. Overfitting and instability

Rarely can an RL agent fit in multiple environments. Even for deep meta-RL, there is no guarantee that the agent can perform well on new tasks. RL is unstable and very sensitive to the initialization and dynamics of the training process. Each category of algorithms has its own chllenges in this regard. For example, fitted Q (or value) methods with deep network function estimators are typically not contractive, hence no guarantee of convergence. In addition, there are many parameters for stability such as target network delay, replay buffer size, and clipping. Policy gradient methods have a very high variance gradient estimator. Although there have been many methods that try to reduce the variance, most also introduce hyper parameters, so they require many samples and complex baseline. Model-based RL algorithms have to choose their model class and a fitting method. The optimization policy is nontrivial due to backpropagation through time. In contrast, the different hyper parameters in supervised learning will show more or fewer changes in training. Bad luck in RL may mean that the curve of the model will not change for a long time or that the RL methods do not work at all. What is worse, even if all the hyper parameters and the random seeds are known, the performance will be very different as long as the implementation is slightly different.

We believe that there are great prospects for RL and list some plausible directions for the future:

1. Algorithms with favorable improvement and convergence are needed. Although TRPO gives a kind of guaranteed improvement under assumptions, it probably does not hold in the real world. Although adjusting parameters adaptively (Gu SX et al., 2017b) requires a large number of samples to provide statistical guarantees, it may be a good starting point.

2. Artificially add some supervision signals. In the case of sparse rewards, we can introduce an intrinsic reward or add some auxiliary tasks to increase the exploration ability.

3. IRL can automatically learn the reward function, and imitation learning does not have high requirements for reward functions. Besides, unsupervised or weakly supervised learning of diverse behaviors will avoid rigid supervision like the reward function. These two directions could make up for the disadvantages of RL.

4. Generalize from multi-task learning. Deep meta-RL is increasingly recognized as one of the most likely ways to implement AGI. Recent studies have shown that the performance of agents can be improved by increasing the exploration ability of meta-RL. Combining meta-RL with IRL is also an attractive direction.

7 Summary

Over the past few years, deep RL has become increasingly powerful and important in handling complex problems. In this survey, we conduct a comprehensive review of deep RL algorithms. First, we introduce the model-free and model-based deep RL algorithms to determine the characteristics of RL. Then we summarize several recent advances and divide them into three categories: exploration methods, inverse RL, and transfer RL. We give a detailed review for each category. In terms of applications, we discuss deep RL in robotics, NLP, and computer systems. Finally, we suggest some open challenges that indicate future research directions for deep RL.

Contributors

Hao-nan WANG designed the research. Ning LIU and Yi-yun ZHANG collected the literature. Hao-nan WANG drafted the manuscript. Ning LIU, Da-wei FENG, and Feng HUANG helped organize the manuscript. Hao-nan WANG and Ning LIU revised the manuscript. Hao-nan WANG finalized the paper under the guidance of Dong-sheng LI and Yi-ming ZHANG.

Compliance with ethics guidelines

Hao-nan WANG, Ning LIU, Yi-yun ZHANG, Da-wei FENG, Feng HUANG, Dong-sheng LI, and Yi-ming ZHANG declare that they have no conflict of interest.

References

Abbeel P, Ng AY, 2004. Apprenticeship learning via inverse reinforcement learning. Proc 21st Int Conf on Machine Learning, p.1-8. https://doi.org/10.1145/1015330.1015430

Achiam J, Held D, Tamar A, et al., 2017. Constrained policy optimization. Proc 34th Int Conf on Machine Learning, p.22-31.

Al-Nima RRO, Han TT, Chen TL, 2019. Road tracking using deep reinforcement learning for self-driving car applications. Int Conf on Computer Recognition Systems, p.106-116. https://doi.org/10.1007/978-3-030-19738-4_12

Arik SO, Chen JT, Peng KN, et al., 2018. Neural voice cloning with a few samples. Proc 32nd Neural Information Processing Systems, p.10019-10029.

Aytar Y, Pfaff T, Budden D, et al., 2018. Playing hard exploration games by watching YouTube. Proc 32nd Neural Information Processing Systems, p.2930-2941.

Bellemare MG, Naddaf Y, Veness J, et al., 2013. The Arcade learning environment: an evaluation platform for general agents. *J Artif Intell Res*, 47:253-279. https://doi.org/10.1613/jair.3912

Bellemare MG, Srinivasan S, Ostrovski G, et al., 2016. Unifying count-based exploration and intrinsic motivation. Proc 30th Neural Information Processing Systems, p.1471-1479.

Bellemare MG, Dabney W, Munos R, 2017. A distributional perspective on reinforcement learning. Proc 34th Int Conf on Machine Learning, p.449-458.

Blundell C, Cornebise J, Kavukcuoglu K, et al., 2015. Weight uncertainty in neural networks. Proc 32nd Int Conf on Machine Learning, p.1613-1622.

Botvinick M, Ritter S, Wang JX, et al., 2019. Reinforcement learning, fast and slow. *Trends Cogn Sci*, 23(5):408-422. https://doi.org/10.1016/j.tics.2019.02.006

Buckman J, Hafner D, Tucker G, et al., 2018. Sample-efficient reinforcement learning with stochastic ensemble value expansion. Proc 32nd Neural Information Processing Systems, p.8224-8234.

Burda Y, Edwards H, Pathak D, et al., 2019. Large-scale study of curiosity-driven learning. https://arxiv.org/abs/1808.04355

Chapelle O, Li LH, 2011. An empirical evaluation of Thompson sampling. Proc 24th Neural Information Processing Systems, p.2249-2257.

Chebotar Y, Hausman K, Zhang M, et al., 2017. Combining model-based and model-free updates for trajectory-centric reinforcement learning. Proc 34th Int Conf on Machine Learning, p.703-711.

Chen L, Lingys J, Chen K, et al., 2018. AuTO: scaling deep reinforcement learning for datacenter-scale automatic traffic optimization. Proc Conf of the ACM Special Interest Group on Data Communication, p.191-205. https://doi.org/10.1145/3230543.3230551

Chen YT, Assael Y, Shillingford B, et al., 2019. Sample efficient adaptive text-to-speech. https://arxiv.org/abs/1809.10460

Chua K, Calandra R, McAllister R, et al., 2018. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. Proc 32nd Neural Information Processing Systems, p.4754-4765.

Devin C, Gupta A, Darrell T, et al., 2017. Learning modular neural network policies for multi-task and multi-robot transfer. Proc IEEE Int Conf on Robotics and Automation, p.2169-2176. https://doi.org/10.1109/ICRA.2017.7989250

Dhingra B, Li LH, Li XJ, et al., 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. Proc 55th Annual Meeting of the Association for Computational Linguistics, p.484-495. https://doi.org/10.18653/v1/P17-1045

Duan Y, Schulman J, Chen X, et al., 2017. RL^2: fast reinforcement learning via slow reinforcement learning. https://arxiv.org/abs/1611.02779

Ebert F, Finn C, Lee AX, et al., 2017. Self-supervised visual planning with temporal skip connections. Proc 1st Annual Conf on Robot Learning, p.344-356.

Feinberg V, Wan A, Stoica I, et al., 2018. Model-based value estimation for efficient model-free reinforcement learning. https://arxiv.org/abs/1803.00101

Finn C, Levine S, 2017. Deep visual foresight for planning robot motion. Proc IEEE Int Conf on Robotics and Automation, p.2786-2793. https://doi.org/10.1109/ICRA.2017.7989324

Finn C, Levine S, Abbeel P, 2016a. Guided cost learning: deep inverse optimal control via policy optimization. Proc 33rd Int Conf on Machine Learning, p.49-58.

Finn C, Tan XY, Duan Y, et al., 2016b. Deep spatial autoencoders for visuomotor learning. Proc IEEE Int Conf on Robotics and Automation, p.512-519. https://doi.org/10.1109/ICRA.2016.7487173

Finn C, Abbeel P, Levine S, 2017a. Model-agnostic meta-learning for fast adaptation of deep networks. Proc 34th Int Conf on Machine Learning, p.1126-1135.

Finn C, Yu T, Zhang T, et al., 2017b. One-shot visual imitation learning via meta-learning. Proc 1st Conf on Robot Learning, p.357-368.

Fortunato M, Azar MG, Piot B, et al., 2019. Noisy networks for exploration. https://arxiv.org/abs/1706.10295

Fu J, Levine S, Abbeel P, 2016. One-shot learning of manipulation skills with online dynamics adaptation and neural network priors. Proc IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.4019-4026. https://doi.org/10.1109/IROS.2016.7759592

Fu J, Co-Reyes JD, Levine S, 2017a. EX^2: exploration with exemplar models for deep reinforcement learning. Proc 30th Neural Information Processing Systems, p.2577-2587.

Fu J, Luo K, Levine S, 2017b. Learning robust rewards with adversarial inverse reinforcement learning. https://arxiv.org/abs/1710.11248

Fujimoto S, Hoof H, Meger D, 2018. Addressing function approximation error in actor-critic methods. Proc 35th Int Conf on Machine Learning, p.1587-1596.

Gal Y, Hron J, Kendall A, 2017. Concrete dropout. Proc 30th Neural Information Processing Systems, p.3581-3590.

Garcia FM, Thomas PS, 2019. A meta-MDP approach to exploration for lifelong reinforcement learning. Proc 32nd Neural Information Processing Systems, p.5691-5700.

Ghasemipour SKS, Gu SX, Zemel R, 2019. SMILe: scalable meta inverse reinforcement learning through context-conditional policies. Proc 32nd Neural Information Processing Systems, p.7879-7889.

Gu JT, Hassan H, Devlin J, et al., 2018a. Universal neural machine translation for extremely low resource languages. Proc 16th Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, p.344-354. https://doi.org/10.18653/v1/N18-1032

Gu JT, Wang Y, Chen Y, et al., 2018b. Meta-learning for low-resource neural machine translation. Proc Conf on Empirical Methods in Natural Language Processing, p.3622-3631. https://doi.org/10.18653/v1/D18-1398

Gu SX, Lillicrap T, Sutskever I, et al., 2016. Continuous deep Q-learning with model-based acceleration. Proc 33rd Int Conf on Machine Learning, p.2829-2838.

Gu SX, Holly E, Lillicrap T, et al., 2017a. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. Proc IEEE Int Conf on Robotics and Automation, p.3389-3396. https://doi.org/10.1109/ICRA.2017.7989385

Gu SX, Lillicrap T, Ghahramani Z, et al., 2017b. Q-Prop: sample-efficient policy gradient with an off-policy critic. https://arxiv.org/abs/1611.02247

Gupta A, Mendonca R, Liu YX, et al., 2018. Meta-reinforcement learning of structured exploration strategies. Proc 32nd Neural Information Processing Systems, p.5302-5311.

Haarnoja T, Tang HR, Abbeel P, et al., 2017. Reinforcement learning with deep energy-based policies. Proc 34th Int Conf on Machine Learning, p.1352-1361.

Haarnoja T, Zhou A, Abbeel P, et al., 2018. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor. Proc 35th Int Conf on Machine Learning, p.1861-1870.

Hausknecht M, Stone P, 2017. Deep recurrent Q-learning for partially observable MDPs. https://arxiv.org/abs/1507.06527

He D, Xia YC, Qin T, et al., 2016. Dual learning for machine translation. Proc 30th Neural Information Processing Systems, p.820-828.

Heess N, Sriram S, Lemmon J, et al., 2017. Emergence of locomotion behaviours in rich environments. https://arxiv.org/abs/1707.02286

Hessel M, Modayil J, van Hasselt H, et al., 2018. Rainbow: combining improvements in deep reinforcement learning. https://arxiv.org/abs/1710.02298

Ho J, Ermon S, 2016. Generative adversarial imitation learning. Proc 30th Neural Information Processing Systems, p.4565-4573.

Horgan D, Quan J, Budden D, et al., 2018. Distributed prioritized experience replay. https://arxiv.org/abs/1803.00933

Houthooft R, Chen X, Duan Y, et al., 2017. Variational information maximizing exploration. Proc 30th Neural Information Processing Systems, p.1109-1117.

Kakade S, Langford J, 2002. Approximately optimal approximate reinforcement learning. Proc 19th Int Conf on Machine Learning, p.267-274.

Kalashnikov D, Irpan A, Pastor P, et al., 2018. QT-Opt: scalable deep reinforcement learning for vision-based robotic manipulation. Proc 2nd Conf on Robot Learning, p.651-673.

Klein E, Geist M, Piot B, et al., 2012. Inverse reinforcement learning through structured classification. Proc 25th Neural Information Processing Systems, p.1007-1015.

Kolter JZ, Ng AY, 2009. Near-Bayesian exploration in polynomial time. Proc 26th Int Conf on Machine Learning, p.513-520. https://doi.org/10.1145/1553374.1553441

Krizhevsky A, Sutskever I, Hinton GE, 2012. ImageNet classification with deep convolutional neural networks. Proc 25th Neural Information Processing Systems, p.1097-1105.

Kröse BJA, 1995. Learning from delayed rewards. *Robot Auton Syst*, 15(4):233-235. https://doi.org/10.1016/0921-8890(95)00026-C

Lange S, Riedmiller M, Voigtländer A, 2012. Autonomous reinforcement learning on raw visual input data in a real world application. Proc Int Joint Conf on Neural Networks, p.1-8. https://doi.org/10.1109/IJCNN.2012.6252823

Levine S, Koltun V, 2013. Guided policy search. Proc 30th Int Conf on Machine Learning, p.1-9.

Levine S, Wagener N, Abbeel P, 2015. Learning contact-rich manipulation skills with guided policy search. Proc IEEE Int Conf on Robotics and Automation, p.156-163. https://doi.org/10.1109/ICRA.2015.7138994

Levine S, Finn C, Darrell T, et al., 2016. End-to-end training of deep visuomotor policies. *J Mach Learn Res*, 17(1): 1334-1373.

Lillicrap TP, Hunt JJ, Pritzel A, et al., 2016. Continuous control with deep reinforcement learning. Proc 4th Int Conf on Learning Representations, p.2829-2838.

Lin LJ, 1992. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Mach Learn*, 8(3-4):293-321. https://doi.org/10.1007/BF00992699

Mao HZ, Alizadeh M, Menache I, et al., 2016. Resource management with deep reinforcement learning. Proc 15th ACM Workshop on Hot Topics in Networks, p.50-56. https://doi.org/10.1145/3005745.3005750

Mao HZ, Schwarzkopf M, Venkatakrishnan SB, et al., 2019a. Learning scheduling algorithms for data processing clusters. Proc ACM Special Interest Group on Data Communication, p.270-288.

Mao HZ, Negi P, Narayan A, et al., 2019b. Park: an open platform for learning-augmented computer systems. Proc 36th Int Conf on Machine Learning, p.2490-2502.

Mishra N, Rohaninejad M, Chen X, et al., 2018. A simple neural attentive meta-learner. https://arxiv.org/abs/1707.03141

Mnih V, Kavukcuoglu K, Silver D, et al., 2013. Playing Atari with deep reinforcement learning. https://arxiv.org/abs/1312.5602

Mnih V, Kavukcuoglu K, Silver D, et al., 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529-533. https://doi.org/10.1038/nature14236

Mnih V, Badia AP, Mirza M, et al., 2016. Asynchronous methods for deep reinforcement learning. Proc 33rd Int Conf on Machine Learning, p.1928-1937.

Mousavi SS, Schukat M, Howley E, 2018. Deep reinforcement learning: an overview. Proc SAI Intelligent Systems Conf, p.426-440. https://doi.org/10.1007/978-3-319-56991-8_32

Nachum O, Norouzi M, Xu K, et al., 2017a. Bridging the gap between value and policy based reinforcement learning. Proc 31st Neural Information Processing Systems, p.2775-2785.

Nachum O, Norouzi M, Xu K, et al., 2017b. Trust-PCL: an off-policy trust region method for continuous control. https://arxiv.org/abs/1707.01891

Nagabandi A, Kahn G, Fearing RS, et al., 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. IEEE Int Conf on Robotics and Automation, p.7559-7566. https://doi.org/10.1109/ICRA.2018.8463189

Nagabandi A, Clavera I, Liu SM, et al., 2019. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. https://arxiv.org/abs/1803.11347v6

Ng AY, Russell SJ, 2000. Algorithms for inverse reinforcement learning. Proc 17th Int Conf on Machine Learning, p.663-670.

Osband I, Blundell C, Pritzel A, et al., 2016. Deep exploration via bootstrapped DQN. Proc 29th Neural Information Processing Systems, p.4026-4034.

Ostrovski G, Bellemare MG, van den Oord A, et al., 2017. Count-based exploration with neural density models. Proc 34th Int Conf on Machine Learning, p.2721-2730.

Parisotto E, Ba JL, Salakhutdinov R, 2016. Actor-Mimic: deep multitask and transfer reinforcement learning. https://arxiv.org/abs/1511.06342

Pathak D, Agrawal P, Efros AA, et al., 2017. Curiosity-driven exploration by self-supervised prediction. Proc IEEE Conf on Computer Vision and Pattern Recognition Workshops, p.488-489. https://doi.org/10.1109/CVPRW.2017.70

Peng XB, Abbeel P, Levine S, et al., 2018a. DeepMimic: example-guided deep reinforcement learning of physics-based character skills. *ACM Trans Graph*, 37(4):143. https://doi.org/10.1145/3197517.3201311

Peng XB, Andrychowicz M, Zaremba W, et al., 2018b. Sim-to-real transfer of robotic control with dynamics randomization. Proc IEEE Int Conf on Robotics and Automation, p.3803-3810. https://doi.org/10.1109/ICRA.2018.8460528

Ping W, Peng KN, Gibiansky A, et al., 2018. Deep voice 3: 2000-speaker neural text-to-speech. Proc Int Conf on Learning Representations, p.214-217.

Pohlen T, Piot B, Hester T, et al., 2018. Observe and look further: achieving consistent performance on Atari. https://arxiv.org/abs/1805.11593

Racanière S, Weber T, Reichert DP, et al., 2017. Imagination-augmented agents for deep reinforcement learning. Proc 31st Neural Information Processing Systems, p.5694-5705.

Rahmatizadeh R, Abolghasemi P, Behal A, et al., 2016. Learning real manipulation tasks from virtual demonstrations using LSTM. https://arxiv.org/abs/1603.03833v2

Rajeswaran A, Ghotra S, Ravindran B, et al., 2017. EPOpt: learning robust neural network policies using model ensembles. https://arxiv.org/abs/1610.01283

Rakelly K, Zhou A, Quillen D, et al., 2019. Efficient off-policy meta-reinforcement learning via probabilistic context variables. Proc 36th Int Conf on Machine Learning, p.5331-5340.

Ratliff ND, Bagnell JA, Zinkevich MA, 2006. Maximum margin planning. Proc 23rd Int Conf on Machine Learning, p.729-736. https://doi.org/10.1145/1143844.1143936

Russo D, Roy BV, 2014. Learning to optimize via information-directed sampling. Proc 27th Neural Information Processing Systems, p.1583-1591.

Rusu AA, Colmenarejo SG, Gulcehre C, et al., 2016a. Policy distillation. https://arxiv.org/abs/1511.06295

Rusu AA, Rabinowitz NC, Desjardins G, et al., 2016b. Progressive neural networks. https://arxiv.org/abs/1606.04671

Schaul T, Quan J, Antonoglou I, et al., 2016. Prioritized experience replay. https://arxiv.org/abs/1511.05952

Schulman J, Levine S, Moritz P, et al., 2015. Trust region policy optimization. Proc Int Conf on Machine Learning, p.1889-1897.

Schulman J, Moritz P, Levine S, et al., 2016. High-dimensional continuous control using generalized advantage estimation. https://arxiv.org/abs/1506.02438

Schulman J, Wolski F, Dhariwal P, et al., 2017. Proximal policy optimization algorithms. https://arxiv.org/abs/1707.06347

Shum HY, He XD, Li D, 2018. From Eliza to XiaoIce: challenges and opportunities with social chatbots. *Front Inform Technol Electron Eng*, 19(1):10-26. https://doi.org/10.1631/FITEE.1700826

Silver D, Lever G, Heess N, et al., 2014. Deterministic policy gradient algorithms. Proc 31st Int Conf on Machine Learning, p.387-395.

Silver D, Huang A, Maddison CJ, et al., 2016. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484-489. https://doi.org/10.1038/nature16961

Skerry-Ryan RJ, Battenberg E, Xiao Y, et al., 2018. Towards end-to-end prosody transfer for expressive speech synthesis with Tacotron. Int Conf on Machine Learning, p.4693-4702.

Stadie BC, Yang G, Houthooft R, et al., 2018. Some considerations on learning to explore via meta-reinforcement learning. https://arxiv.org/abs/1803.01118

Strehl AL, Littman ML, 2008. An analysis of model-based interval estimation for Markov decision processes. *J Comput Syst Sci*, 74(8):1309-1331. https://doi.org/10.1016/j.jcss.2007.08.009

Sutton RS, 1988. Learning to predict by the methods of temporal differences. *Mach Learn*, 3(1):9-44. https://doi.org/10.1023/A:1022633531479

Sutton RS, Barto AG, 2018. Reinforcement Learning: an Introduction (2nd Ed.). MIT Press, Cambridge, MA, USA.

Tang HR, Houthooft R, Foote D, et al., 2017. #Exploration: a study of count-based exploration for deep reinforcement learning. Proc 31st Neural Information Processing Systems, p.2753-2762.

van Hasselt H, Guez A, Silver D, 2016. Deep reinforcement learning with double Q-learning. Proc 30th AAAI Conf on Artificial Intelligence, p.2096-2100.

Vanschoren J, 2018. Meta-learning: a survey. https://arxiv.org/abs/1810.03548

Vinyals O, Ewalds T, Bartunov S, et al., 2017. StarCraft II: a new challenge for reinforcement learning. https://arxiv.org/abs/1708.04782

Wang JX, Kurth-Nelson Z, Tirumala D, et al., 2017. Learning to reinforcement learn. https://arxiv.org/abs/1611.05763

Wang ZY, Schaul T, Hessel M, et al., 2016. Dueling network architectures for deep reinforcement learning. Proc 33rd Int Conf on Machine Learning, p.1995-2003.

Wang ZY, Bapst V, Heess N, et al., 2017. Sample efficient actor-critic with experience replay. https://arxiv.org/abs/1611.01224

Watter M, Springenberg JT, Boedecker J, et al., 2015. Embed to control: a locally linear latent dynamics model for control from raw images. Proc 28th Neural Information Processing Systems, p.2746-2754.

Williams RJ, 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach Learn*, 8(3-4):229-256. https://doi.org/10.1023/A:1022672621406

Wu YH, Mansimov E, Grosse RB, et al., 2017. Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation. Proc 30th Neural Information Processing Systems, p.5279-5288.

Xia C, El Kamel A, 2016. Neural inverse reinforcement learning in autonomous navigation. *Robot Auton Syst*, 84:1-14. https://doi.org/10.1016/j.robot.2016.06.003

Yahya A, Li A, Kalakrishnan M, et al., 2017. Collective robot reinforcement learning with distributed asynchronous guided policy search. IEEE/RSJ Int Conf on Intelligent Robots and Systems, p.79-86. https://doi.org/10.1109/IROS.2017.8202141

Yu TH, Finn C, Xie AN, et al., 2018. One-shot imitation from observing humans via domain-adaptive meta-learning. https://arxiv.org/abs/1802.01557v1

Yu WH, Tan J, Liu CK, et al., 2017. Preparing for the unknown: learning a universal policy with online system identification. https://arxiv.org/abs/1702.02453

Zhang M, Vikram S, Smith L, et al., 2019. SOLAR: deep structured representations for model-based reinforcement learning. Proc 36th Int Conf on Machine Learning, p.7444-7453.

Ziebart BD, Maas A, Bagnell JA, et al., 2008. Maximum entropy inverse reinforcement learning. Proc 23rd AAAI Conf on Artificial Intelligence, p.1433-1438.

Zintgraf L, Shiarli K, Kurin V, et al., 2019. Fast context adaptation via meta-learning. Proc 36th Int Conf on Machine Learning, p.7693-7702.