IEICE TRANS. INF. & SYST., VOL.E103–D, NO.9 SEPTEMBER 2020

PAPER

Hybrid of Reinforcement and Imitation Learning for Human-Like Agents

Rousslan F. J. DOSSA[†a)], **Xinyu LIAN**[†b)], **Hirokazu NOMOTO**[††c)], ***Nonmembers***, **Takashi MATSUBARA**[†††d)], ***and*** **Kuniaki UEHARA**[††††e)], ***Members***

SUMMARY Reinforcement learning methods achieve performance superior to humans in a wide range of complex tasks and uncertain environments. However, high performance is not the sole metric for practical use such as in a game AI or autonomous driving. A highly efficient agent performs greedily and selfishly, and is thus inconvenient for surrounding users, hence a demand for human-like agents. Imitation learning reproduces the behavior of a human expert and builds a human-like agent. However, its performance is limited to the expert's. In this study, we propose a training scheme to construct a human-like and efficient agent via mixing reinforcement and imitation learning for discrete and continuous action space problems. The proposed hybrid agent achieves a higher performance than a strict imitation learning agent and exhibits more human-like behavior, which is measured via a human sensitivity test.

key words: *autonomous driving, game AI, human-like behavior, imitation learning, reinforcement learning*

1. Introduction

Reinforcement learning (RL) achieved impressive progress in several areas including Go [1], autonomous driving [2]–[5], and video games [6], [7]. The RL trains an agent to maximize future rewards, and the trained agent occasionally outperforms human experts. However, high efficiency is not the sole purpose for practical use. When a non-player character (NPC) is trained by RL, it can become too strong to be defeated by human players, and the players are quickly frustrated and are unable to enjoy the game. Similarly, an efficient autonomous vehicle can inconvenience surrounding cars and pedestrians by taking abrupt turns or more generally by exhibiting selfish behavior or creating a sense of fear even when the behavior is efficient and safe in reality. Hence, an approach to build an agent that behaves like a human is desirable.

Manuscript received November 8, 2019.
Manuscript revised April 27, 2020.
Manuscript publicized June 15, 2020.
[†]The authors are with Graduate School of System Informatics, Kobe University, Kobe-shi, 657–8501 Japan.
[††]The author is with EQUOS RESEARCH Co., Ltd., Tokyo, 101–0021 Japan.
[†††]The author is with Graduate School of Engineering Science, Osaka University, Toyonaka-shi, 560–8531 Japan.
[††††]The author is with Faculty of Business Administration, Osaka Gakuin University, Suita-shi, 564–8511 Japan.
a) E-mail: doss@ai.cs.kobe-u.ac.jp
b) E-mail: rensy94@ai.cs.kobe-u.ac.jp
c) E-mail: i19341_nomoto@aisin-aw.co.jp
d) E-mail: matsubara@sys.es.osaka-u.ac.jp
e) E-mail: kuniaki.uehara@ogu.ac.jp
DOI: 10.1587/transinf.2019EDP7298

Imitation learning (IL) trains an agent to learn a policy from training data provided by a human expert [8], [9]. An agent trained by IL is expected to exhibit human-like behavior. However, IL is not a goal-oriented method and is dependent on the demonstration data provided by a human expert. Hence, the performance of an IL agent is likely to be capped to the former's. Various studies have leveraged human expert demonstrations to improve the performance of the RL agent [10]–[13]. However, there is a paucity of studies on the human-likeliness factor.

In the study, we aim to produce an agent that behaves like a human while retaining high performance by RL. We proposed a hybrid agent based on RL and IL and demonstrate how the performance level of an RL agent and tendency of human expert behavior can be transferred via IL to a student agent. We applied our approach to an original game and the Atari57 suite's Gopher game [14] as environments of discrete action spaces. We also applied our approach to the TORCS racing simulator [15] as an environment of a continuous action space to demonstrate its potential in real-life applications, such as autonomous driving, where a human-like agent is highly desirable. A performance test and sensitivity test (like the "Turing test") conducted in a double-blind fashion demonstrated that the proposed approach built agents that achieve better performance and exhibit more human-like behaviors when compared to those of IL and RL agents, respectively.

The structure of this paper is as follows: we first review the existing works in the field in Sect. 2, then define the basics of the methods which were built upon as the proposed method in Sect. 3. We describe the experiments conducted using the proposed method, followed by the corresponding results and discussions in Sect. 4. Finally, we conclude in Sect. 5.

Preliminary and limited results are given in the conference proceedings [16].

2. Related Work

2.1 Improving Reinforcement Learning Performance from Expert Demonstrations

Across several topics such as robotics, general games, or autonomous driving, several studies have aimed to combine reinforcement learning (RL) and imitation learning (IL). The precursors of the aforementioned studies include

Chang et al. [10] who proposed locally optimal learning to search (LOLS) intended for structured prediction tasks. This is based on the more general learning to search (L2S) method [8], [17]–[19]. Specifically, LOLS involves randomly sampling from either the learning policy or reference policy (expert policy) given an arbitrary state during the rollouts. Subsequently, it minimizes the objective loss function and guarantees convergence to a policy that is at least as good as the reference policy.

Following this, Nair et al. [11] extended the concept via the usage of expert demonstrations to overcome the difficulties of exploration in tasks with high dimensional state-action space or sparse rewards. They introduced a demonstration buffer that stores trajectories sampled from an expert, which are subsequently used in combination with the trajectories sampled from the learning policy to update the value function. Thus, the learning policy is oriented toward more promising avenues and consequently speeds up the training process. The authors demonstrated the effectiveness of their method by applying it to a 7-DOF fetch robotic arm to solve a block stacking problem with a continuous action space. The proposed method achieved a relatively better exploration when compared to the baselines. It also exhibited a performance exceeding the expert's performance in solving the stacking task when provided with a few additional simulator tweaks.

Similarly, Hester et al. [12] proposed deep Q-Learning from demonstrations (DQfD) that also leverages a very small amount of demonstration data to significantly accelerate the training process. The agent is initially pretrained on the expert's demonstration data via a combination of both supervised and temporal difference (TD) losses based on the vanilla deep Q-learning (DQN) [6] and is subsequently trained by interacting with the domain with its pretrained policy as usual. Hence, the agent learns to imitate the expert's policy while improving it with the self-learned policy via RL. The resultant policy outperforms pure reinforcement learning baseline (double dueling DQN [20]) in 41 out of the 42 games that are experimented on with a significantly faster convergence rate.

More recently, Balaguer et al. [21] focused on a robotic arm problem, namely towel folding with cooperative manipulators through momentum fold. Their proposed algorithm leveraged human demonstrations to reduce the search space of the RL learner, thereby allowing it to converge faster to a satisfactory policy. Furthermore, Sun et al. [22] introduced Truncated HORizon Policy Search (THOR), which is based on the idea of reshaping the reward [23] and leveraged even sub-optimal expert demonstrations to shorten the learner's planning horizon, thereby significantly accelerating the learning process. Additionally, their algorithm keeps track of a disadvantage function between the expert and learner policy and prioritizes the optimization of the self-learned policy when the policy induced by the expert is determined as sub-optimal, and thus the final policy outperforms the expert.

Le et al. [13] introduced hierarchically guided IL and RL in the hierarchical reinforcement learning framework (where a high-level policy or meta-controller is trained to select a sub-goal to be solved by low-level sub-policies until the final goal is achieved). By training the meta-controller following IL of a high-level interactive expert, hierarchical guidance limits the training of the RL sub-policies to the relevant state space and consequently improves the sample efficiency of the training process while reducing the cost of expert demonstrations.

The aforementioned studies mainly focus on improving the performance, sampling efficiency, and training speed of the RL agents. By contrast, the present study investigates the feasibility of grasping desirable features to characterize a human expert while improving the performance following an RL agent or self-learned behavior.

2.2 Building Human-Like Behavior Based on Human Demonstrations

To the best of the authors' knowledge, a conceptually close work was only explored by Hecker et al. [24]. They focused on autonomous driving and proposed a formalism to achieve an accurate and comfortable human-like driving. The method achieves a better driving accuracy by augmenting the data set consisting of car-mounted front-facing images data of human driving with a manually engineered set of map features. The learning policy also incorporates long-short term neural networks to consider previously executed maneuvers, and thus its output is sequential as opposed to traditional the point-wise predictions. Furthermore, the driving loss is enhanced with an adversarial loss that discriminates between human-like driving and policy learning via RL in addition to a loss that penalizes abrupt steering or acceleration over a short period, thereby resulting in a final policy that is close to the human expert's demonstrations. However, the human-likeliness of the behavior is systematically evaluated by clustering the driving data of the human expert and learned policy's predictions and measuring their similarity. Although such an evaluation method is pragmatic, it does not guarantee a measurement of human impression, namely because they are equivalent to the metrics used for imitation learning of a human expert.

3. Methods

3.1 Preliminaries

3.1.1 Reinforcement Learning

We introduce the reinforcement learning (RL) via the framework of Markov decision processes (MDP). An MDP is defined as a 5-tuple $< \mathcal{S}, \mathcal{A}, P, R, \gamma >$, where $\mathcal{S}$ denotes a set of states, $\mathcal{A}$ denotes a set of actions, $P_{s'}(s,a) = P(s_{t+1} = s' | s_t = s, a_t = a)$ denotes the probability that state s transits to state s' due to action a on time-step t, $R(s,a)$ denotes the reward received after the transition from state s to the state

s', and γ $(0 < \gamma \leq 1)$ denotes a discount factor that represents the importance of future rewards when compared to the present rewards. On every time-step t during training, the agent obverses a state $s_t \in \mathcal{S}$ and performs an action $a_t \in \mathcal{A}$. In response, the environment returns the corresponding reward r_t and next state s_{t+1}.

With respect to a discrete action space case, the deep Q-network (DQN) algorithm [6] achieved superhuman performance in numerous experiments. From an arbitrary state s_t at time-step t, the agent is trained to predict a future return $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$ where T denotes the terminal time-step. The agent subsequently takes an action that maximizes R_t based on the prediction. The optimal action-value function is defined as $Q^*(s_t, a_t) = \max_\pi \mathbb{E}[R_t | s = s_t, a = a_t]$, and applying the Bellman equation to this yields the following expression:

$$Q^*(s_t, a_t) = \mathbb{E}_{s_{t+1} \sim \mathcal{E}}[r_t + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) | s_t, a_t]$$

where $\mathcal{E}$ represents the transition function of the environment. During training, the agent estimates R_t by iteratively updating the action-value function $Q(\phi(s), a)$ where ϕ denotes a function that produces the fixed length representation of state s. Thus, the action-value function $Q(\phi(s), a)$ converges to the optimal $Q_i \rightarrow Q^*$ as the update iteration $i \rightarrow \infty$ [25].

With respect to a continuous action space case, we redefine the action space as $\mathcal{A} \subset \mathbb{R}^N$ to be continuous and set the goal to obtain a policy π that maximizes the expected return from the start distribution $J = \mathbb{E}_{r_i, s_i \sim \mathcal{E}, a_i \sim \pi}[R_0]$. Although DQN achieves strong performance over a high dimensional state space, it was proven as limited to low dimension discrete action spaces. The deep deterministic policy gradient (DDPG) method [26] is based on the deterministic policy gradient (DPG) method [27], which defines the policy of the RL agent as an actor function $(\mu(s|\theta^\mu)$ parametrized by θ^μ and mapped from a state s_t to a corresponding action $a_t)$ and a critic function $Q(s_t, a_t)$ to approximate the value of a state-action pair (s_t, a_t). The actor update is subsequently performed based on the *policy gradient* as defined by David Silver et al. [27].

The introduction of deep and non-linear approximators makes the update of the critic function $Q(s, a|\theta^Q)$ prone to divergence. Thus, the DDPG method leverages "soft" target updates by creating copies $\mu'(s|\theta^{\mu'})$ and $Q'(s, a|\theta^{Q'})$ to be used in computing the target values. However, the target networks are updated by slowly tracking original networks: $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$ with $\tau << 1$ to increase the stability of the training.

Given that a large action space requires considerably more exploration to converge to the optimal solution, an exploration policy $\mu'(s_t) = \mu(s_t|\theta_t^\mu) + \mathcal{N}$ with an added noise sampled from a noise process $\mathcal{N}$ (which is selected based on the nature of the task or the environment [28]) is adopted.

3.1.2 Imitation Learning

We assume that the policy followed by the expert players during demonstrations corresponds to the optimal policy π^*, and the agent learns a policy π that approximates the optimal policy π^*. A traditional approach involves training an agent via supervised learning. Expert human players provide trajectories that consist of sequences of state-action pairs because the training data follows an optimal policy π^*. The agent's policy is trained to predict actions based on the provided states and thereby imitates the behavior of the expert.

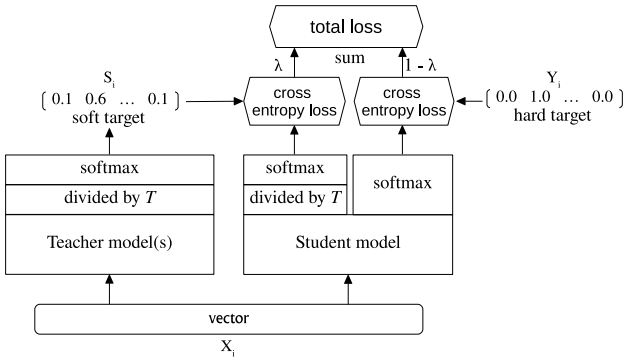

Fig. 1 Procedure of knowledge distillation.

In the discrete action space case, knowledge distillation is an approach that trains a student model based on teacher model(s) where the teacher model trains the student model by using *soft labels* instead of the one-hot label commonly used in traditional supervised learning [29]. When compared to the one-hot label, the values of each dimension of a soft label include more information. With a visual input of a cat, of course, the probability of it being classified as "cat" by the model is the highest, and the probability of "dog" exceeds that of "carrot", and this matches with the fact a cat is visually more similar to a dog than to a carrot. The procedure of knowledge distillation is shown in Fig. 1. The approach achieves a higher performance model with less training data, and it is also useful in terms of compressing large models. An extant study [30] extended the approach to reinforcement learning area, and this is termed as *policy distillation*. Thus, a student model that achieves higher performance is successfully trained by distilling the policy of the teacher model. Furthermore, the applicability of policy distillation to multi-task problems is demonstrated by distilling policies of teacher models trained for different tasks.

Furthermore, in the continuous action space case, generative adversarial imitation learning (GAIL) [31] is an IL method rooted in both inverse reinforcement learning (IRL) and classical RL. Specifically, the IRL initially fits a cost function c from a family of functions C which assigns a low cost to the expert policy π_E and high cost to the other policies π:

$$\max_{c \in C} \left(\min_{\pi \in \Pi} -H(\pi) + \mathbb{E}_\pi [c(s, a)] \right) - \mathbb{E}_{\pi_E} [c(s, a)]$$

where Π denotes the set of all stationary stochastic policies mapping from $\mathcal{S}$ to $\mathcal{A}$, and $H(\pi) \triangleq \mathbb{E}_\pi[-\log \pi(a|s)]$ denotes the γ-discounted causal entropy of the policy π [31]. Con-

versely, classical RL determines the optimal policy from the cost function c extracted by IRL

$$RL(c) = \underset{\pi \in \Pi}{\operatorname{argmin}} -H(\pi) + \mathbb{E}_{\pi}[c(s,a)]$$

corresponding to the expert policy π_E. To bypass the intermediate IRL step, the original study introduced the concept of occupancy measure ρ_π of a policy π, and this is interpreted as the distribution of state-action pairs that an agent encounters while navigating the environment by the following said policy [31]. A function approximator $D_w : \mathcal{S} \times \mathcal{A} \rightarrow (0, 1)$ is used to discriminate between the occupancy measure ρ_π of the student policy and expert ρ_{π_E} by minimizing the sum of their respective expectations over generated trajectories:

$$\mathcal{L}(\pi, \pi_E) = \mathbb{E}_{\pi}[\log(D_w(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D_w(s,a))] - \lambda H(\pi) \text{ for } \lambda \geq 0$$

until D_w is unable to distinguish between the π and π_E, thereby implying that the student learned the expert's policy.

The training of the student agent itself is based on Trust Region Policy Optimization (TRPO) algorithm introduced in [32] where instead of the classical reward, the algorithm is provided with the similarity between the trajectories resulting from the student's policy and that of the expert. More specifically, the student agent is updated using the TRPO rule with the cost function $\log(D_w(s,a))$ by taking a KL-constrained gradient step with respect to

$$\mathbb{E}_{\tau \sim \pi}[\nabla_\theta \log \pi(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta),$$
$$\text{where } Q(\bar{s}, \bar{a}) = \mathbb{E}_{\tau \sim \pi}[\log(D_w(s,a)) | s_0 = \bar{s}, a_0 = \bar{a}].$$

3.2 Proposed Hybrid Agent

Our purpose is to build an agent with a behavior similar to a human expert while retaining the high performance of an RL agent. The problem is separated in the two following subproblems: (1) building an agent that exhibits a superhuman performance and (2) building an agent that selects actions similar to a human expert. The two subproblems are tackled by RL and IL, respectively, and thus we consider our task as a multi-task learning problem. In the study, we propose a method for mixing RL and IL via policy distillation for the discrete action space case and via adversarial imitation learning for the continuous action space case.

Generally, we express the objective function of IL as $\mathcal{L}_{\pi^*}(\pi)$, where π^* denotes a teacher policy to be imitated by the student policy π. Subsequently, the proposed objective function is as follows:

$$\mathcal{L}_{mix}(\pi; \pi_{RL}, \pi_{HE}) = \alpha \mathcal{L}_{\pi_{RL}}(\pi) + (1 - \alpha)\mathcal{L}_{\pi_{HE}}(\pi),$$

where π_{RL} denotes an optimal policy built by RL, π_{HE} denotes a policy of a human expert (HE), and α denotes a trade-off coefficient between the RL policy and human expert.

3.2.1 Discrete Action Space

In the case of discrete action space, we define the objective function of IL as the following cross-entropy loss

$$\mathcal{L}_{\pi^*}(\pi) = \mathbb{E}_s\left[-\sum_a \pi^*(a|s) \log \pi(a|s)\right],$$

by following previous studies on IL and distillation [29], [30]. An RL policy π_{RL} is expressed as a mathematically modeled probability while the policy π_{HE} of a human expert is given in terms of empirical demonstrations. Previous studies on distillation demonstrated that a student model improves its performance by a large margin if it reproduces soft labels (i.e., probability of teacher's output obtained via a high temperature T) in addition to hard labels (i.e., correct labels expressed by 0 or 1 probabilities). It is not possible to obtain soft labels of the policy for distillation. Given this, we used human expert demonstrations π_{HE} as hard labels and the policy $\pi_{RL}^{(T)}$ of a DQN model [6] for soft labels with a temperature T while a typical policy of a DQN model employs $T = 0.0$ (i.e., provides hard labels). Subsequently, the final objective function is expressed as follows:

$$\mathcal{L}_{mix}(\pi) = \alpha\, \mathbb{E}_s\left[-\sum_a \pi_{\mathrm{RL}}^{(T)}(a|s) \log \pi(a|s)\right] + (1 - \alpha)\, \mathbb{E}_s\left[-\sum_a \pi_{HE}(a|s) \log \pi(a|s)\right].$$

It should be noted that the distribution of state s over which the objective is averaged depends on the trajectories sampled from policies. Naturally, the RL policy π_{RL} and human expert policy π_{HE} are used for the first and second terms, respectively. However, when the two policies are completely different, the student policy π encounters a high variance between their actions and yields poor results. We used the human expert policy π_{HE} for the sampling state s for both terms and confirmed that the sampling strategy improves performance and yields a consistent behavior.

3.2.2 Continuous Action Space

While policy distillation can theoretically be applied to continuous action space task, depending on the formulation of the action space itself, simply mixing the actions provided multiple expert is likely to induce a wrong hybrid policy. Namely in the autonomous driving task, a common formulation would be as such: the action of *steering left* is bounded to -1.0, *steering right* is bounded to 1.0, while the *no steering* is bound to 0.0. Let's next assume we want to learn a hybrid of two expert policies when avoiding an obstacle, but with one expert avoiding said obstacle by steering left, while the second one avoids the obstacle by steering right. By mixing those two policies using policy distillation as in the discrete action space case, the hybrid policy is bound to going straight ahead, thus crashing into the obstacle and resulting in a failure. We instead aim to imitate the hybrid of

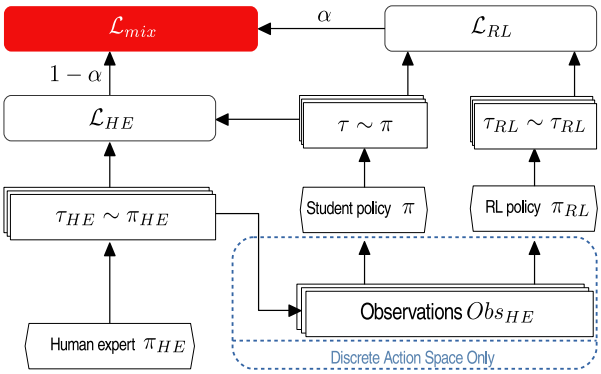

Fig. 2 Conceptual diagram of the proposed hybrid loss.

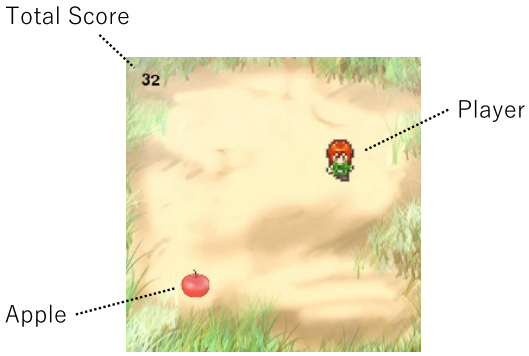

Fig. 3 Screenshot of the Apple game.

the two expert policies, but based on their respective probability density distribution over the state-action space, therefore motivating our choice of the GAIL method introduced in Sect. 3.1.2.

GAIL requires empirical trajectories $\tau^* \sim \pi^*$ obtained from a teacher policy π^* for training. The objective function of the discriminator that is to be maximized by D_w and to be minimized by a student policy π is as follows:

$$\mathcal{L}_{\pi^*}(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\log\left(D_w(s,a)\right)\right] + \mathbb{E}_{\tau^*\sim\pi^*}\left[\log\left(1-D_w(s,a)\right)\right],$$

where τ denotes trajectories $\tau \sim \pi$ sampled from a student policy π. With a human expert and an RL agent providing trajectories $\tau_{HE} \sim \pi_{HE}$ and $\tau_{RL} \sim \pi_{RL}$, respectively, the hybrid loss function is expressed as follows:

$$\begin{aligned}\mathcal{L}_{mix}(\pi) = {} & \mathbb{E}_{\tau\sim\pi}\left[\log\left(D_w(s,a)\right)\right] \\ & + \alpha\,\mathbb{E}_{\tau_{RL}\sim\pi_{RL}}\left[\log\left(1-D_w(s,a)\right)\right] \\ & + (1-\alpha)\,\mathbb{E}_{\tau_{HE}\sim\pi_{HE}}\left[\log\left(1-D_w(s,a)\right)\right], \quad (1)\end{aligned}$$

Intuitively, given that the discriminator trained to recognize an *intermediate* hybrid policy between the RL and human expert policies, the student model — which is trained to learn a policy to fool the discriminator to classify its actions as being the expert's — is expected to converge to the said hybrid policy, thereby exhibiting behaviors inherited from both experts. In contrast to the discrete action space case, the results indicated that the proposed method successfully trained a hybrid policy although the demonstration sets of both experts are sampled independent of each other. We attribute this to the relatively low variation between the RL and human expert behaviors and the adversarial training framework. Figure 2 recapitulates the complete procedure for both action spaces.

4. Experiments and Results

4.1 Apple Game

4.1.1 Experimental Setting

We first applied our approach to an original game termed as *Apple game*, whose screenshot is provided in Fig. 3. This environment consists of a two-dimensional plane on which the agent can move freely in any direction. Additionally apples are spawned randomly at an unpredictable location for the player to take. Once the player's avatar has collected an apple, a new one will spawn in another location on the two-dimensional plane. In the case an apple is not picked up in a fixed amount of time, it disappears and another apple is spawned at a different location. The score of the agent is increased by one point for every apple it picks up. The objective of the agent is thus to maximize the number of apples picked up in a limited amount of time. In this study, we fixed this interval to 30 seconds. Once an apple is spawned, it is always visible until it disappears. Therefore the agent can easily find a sequence of actions to collect it. This environment was select given the straight forward formulation of the objective and the relative ease of evaluating the optimal trajectory taken by an arbitrary agent, as well as its behavior. The diverse combinations of actions give rise to a plurality of behaviors that can help characterize the agent that is playing. Namely, to reach an apple located in the upper left corner, the agent can sequentially move "up" then "left". The agent can also move diagonally toward the apple by using the combination "up and left". The discrete action space $\mathcal{A}$ consists of 9 actions, i.e. moving in either one of the 8 possible directions in a 2D plane or keeping still. The action space is thus represented by $\mathcal{A} = [(-1,-1),(0,-1),\ldots,(1,1)]$, $|\mathcal{A}| = 9$. From the evaluation perspective, given the location of the apple and the player's avatar, it is easy to determine the shortest possible path. This is especially useful to verify whether or not an agent has converged to an optimal policy.

From the human and RL agent, we collected 16,000 frames of gameplay, which were randomly sampled into training and test sets with sizes corresponding to 14,500 and 1,500, respectively, during the training process to maximize the test accuracy. We trained a DQN model to serve as a baseline for comparison and as the RL teacher. The model was trained over $8 \cdot 10^6$ frames of gameplay, with a replay buffer of size $2 \cdot 10^5$, and an exploration coefficient decaying from 1.0 to 0.1, the other hyperparameters being set to their respective default values as per the OpenAI Baselines imple-

Table 1 Results of Apple game.

	Game Score				Sensitivity Test
Agent	**Max**	**Min**	**Average**	**Std.**	**Identified as Human (out of 50)**
Human	27	11	18.71	2.86	**32**
DQN (RL)	53	15	**36.27**	5.44	4
Behavior Cloning (IL)	29	3	17.57	4.37	<u>27</u>
Proposed hybrid agent (RL+IL)	35	11	<u>22.02</u>	3.70	22

Best and second best results are emphasized in bold fonts and underlines, respectively.
The evaluated hybrid model corresponds to a trade-off coefficient of $\alpha = 0.93$.
The sensitivity test for the Apple game involved 25 participants only.

mentation [33]. The student model was trained using Adam optimizer [34] with a learning rate of 10^{-4} and dropout ratio of 0.5 [35]. Additionally, we performed a preparatory search for the α parameter to determine the most suitable one for the task. The duration of one episode is set to 30 seconds, and the number of apples collected by the player during that time interval is considered as the return of the episode.

In addition to the performance evaluation of each model, we also conducted a sensitivity test in a double-blind fashion to evaluate the human-likeliness of agent behaviors. The test involved 26 participants (23 males and 3 females) wherein their age ranged from 27 to 59 with an average age of 44 years, which had no prior exposure to the research materials, demonstrations, or previous studies. Each participant was initially introduced to the rules of the game and provided with the opportunity to play the game to get familiar with its mechanics and how a human would play. In the Apple game case, each participant was presented with two 15 seconds videos of each model playing the game and requested to label it as either a human or a machine, as well as providing comments which motivated such decision.

4.1.2 Results and Discussion

For the Apple game, the score evaluation process for each agent consisted in computing the average of the scores achieved over 400 episodes. In the Apple game, the score denotes the number of apples collected in a 30-s interval. The score variation of the proposed hybrid agent with respect to the change in the trade-off coefficient α is shown in Fig. 4 with the average scores of the human expert, DQN agent, and imitation of the human expert by using classical supervised learning.

At $\alpha = 0.0$ (which represents the distillation of human expert behavior only), the trained policy achieved lower than average performance, and this subsequently continued to decrease when α increases. With respect to a middle-range trade-off coefficient α, mixing the probability distributions over the action space received from the human expert and RL agent resulted in a flat probability distribution over the action space, thereby making the agent follow a random policy and resulting in a lower performance as shown in Fig. 4. The score subsequently rises progressively from $\alpha = 0.7$ until it reaches its peak at $\alpha = 1.0$, and this was equivalent to a distillation of the DQN agent's policy only.

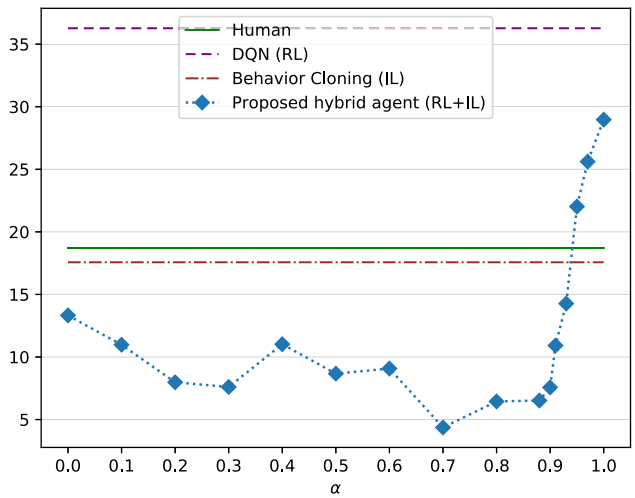

Fig. 4 Effect of the trade-off coefficient α on the proposed hybrid agent's score for the Apple Game. The average scores of the other agents are also provided for reference purposes.

Given the limited availability of the sensitivity test, the representative trade-off coefficient alpha was selected based on the corresponding performance. Following the performance of the hybrid agent obtained across various values of α (Fig. 4), we can see that the proposed method works better with larger values in this environment. We hypothesis this is due to the wide gap in performance between the RL agent and the human expert. More specifically, the gap in performance between the RL agent and the human experts suggests the policies they respective follow are different. Furthermore, given the high variance of the human expert behavior, their policies become hard to mix, thus resulting in a poor hybrid policy. The goal of this study being not just a high performing agent, but also an agent that retains a human-like behavior, we settled on $\alpha = 0.93$ as the smallest experimented value that performs better than the human expert.

Following the results documented in Table 1 and given the simple principle of the game (i.e., moving the player avatar to the spawning apples), the DQN method achieved the highest scores, and this was followed by the proposed method and finally the human agent and its imitation. With respect to the human-likeliness, the human agent was placed first. The proposed hybrid agent exhibited more human-like

Fig. 5 A screenshot of *Gopher*.

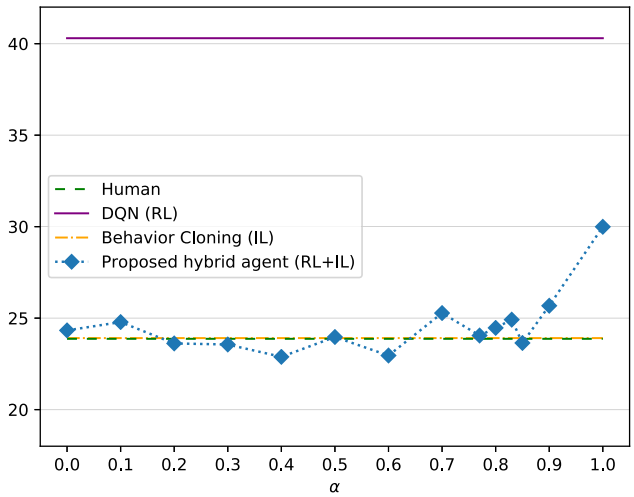

Fig. 6 Effect of the trade-off coefficient α on the proposed hybrid agent's score for Gopher. The average scores of the other agents are also provided for reference purposes.

behavior than the RL agent. It also surpassed the human expert and IL agent in terms of the score and was followed by the human imitation agent. The DQN agent achieved the least human-like behavior.

Additionally, the hybrid agent mitigated a few of the behaviors that gave out the DQN agent, namely a rapid reaction time and mechanical movement towards the apple. This was corroborated by the comments collected from the survey participants including "a moderate reaction span when the apple spawns", "the player takes some time before moving toward the apple", or "keeps going on even after collecting the apple". Similar comments were also used to characterize the behavior of the human expert, thus further illustrating the similar of the proposed hybrid agent with the later.

Hence, we concluded that the proposed approach balances the human-like behavior and exhibited high-performance in the game.

4.2 Atari57 Suite: Gopher Game

4.2.1 Experimental Setting

We also applied our method to the game of the Atari57 suite [14] termed as *Gopher*. The dynamics and the observation space of the Gopher game are more complex than those of the Apple game. This enables us to evaluate the effectiveness of the proposed method in the context of a wider state-action space. An optimal agent in the Gopher game is also expected to act strategically which further raises the complexity of the task. Namely, the goal consists in preventing the "gopher" — a mole-like creature — from digging its way out of the ground and stealing the carrots under the protection of the player, the latter being tasked of moving laterally and filling up the holes as the gopher digs them. A screenshot of the game is shown in Fig. 5. The training data provided by the human expert and RL agent consisted of 55,000 frames each, which were also randomly sampled in training and test sets with sizes of 50,000 and 5,000, respectively. The DQN model that served as an RL teacher was trained over $1.001 \cdot 10^7$ frames of gameplay, a replay buffer of 10^6, and the exploration coefficient decaying from 1.0 to 0.01 while the other hyperparameters remained set to their respective default value as per the OpenAI Baselines implementation [33]. The student model was also trained by following the same method used for the Apple game.

Similarly to the Apple game, we conducted a sensitivity test in a double-blind fashion on top of the performance evaluation of each model. After being introduced to the rules of the game and familiarizing themselves with the game mechanics, each participant was also presented with two 15 seconds videos of each model playing the game and requested to label it as either a human or a machine, as well as providing comments which motivated such decision.

4.2.2 Results and Discussion

We experimented on the Atari57 suite's Gopher game namely by investigating the trade-off coefficient α variation's effect on the performance of the proposed hybrid agent, the results being documented in Fig. 6. The score of the hybrid agent remained around the human expert's average score for $\alpha \in [0.1, 0.6]$ before rising from $\alpha = 0.7$.

As for the Apple game, the gap in performance between the RL agent and the human expert was also high. Henceforth, relatively large values of α are needed to offset the difficulty of mixing the policies that arise given said gap. As a representative of the hybrid agents, we settled for $\alpha = 0.8$, which was the smallest value of α we experimented on that performed better than the human expert.

From a performance viewpoint, the DQN agent obtained the highest score by a large margin. This was followed by the hybrid agent that was trained via the proposed method and finally by the human agent and its imitation.

Table 2 compares the hybrid agent with the others. The hybrid agent (despite prioritizing the RL label by us-

Table 2 Results of Gopher.

	Game Score				Sensitivity Test
Agent	**Max**	**Min**	**Average**	**Std.**	**Identified as Human (out of 52)**
Human	81	2	23.87	19.81	<u>29</u>
DQN (RL)	246	0	**40.30**	36.81	17
Behavior Cloning (IL)	126	0	23.91	23.79	**31**
Proposed hybrid agent (RL+IL)	132	0	<u>26.05</u>	24.31	**31**

Best and second best results are emphasized in bold fonts and underlines, respectively.
The evaluated hybrid model corresponds to a trade-off coefficient of $\alpha = 0.8$

ing $\alpha = 0.8$ to compute the weighted sum of the teachers' labels during the training) only exhibited an improvement of 3 points over the human agent and was still considerably behind the DQN agent's score. The RL agent achieved the best results although only a few participants identified it as a human. In the game, the hybrid agent performs comparatively or superiorly to the IL agent in terms of both criteria, namely game score and human-like behavior. Also, the proposed hybrid agent and IL agent were recognized as humans in the sensitivity test at a similar level to the human agent. This is supported by the comments collected from the participant of the sensitivity test such as "the player seems to move strategically to fill up the holes", "the player follows the movement of the gopher and fills the holes as they are dug out", or "the reaction of the player to the movement of the gopher is slow", which also match the comments characterizing the demonstrations of the human expert.

Meanwhile, the DQN agent was identified as the least human-like by the participants, which transpired in comments such as "the player systematically fills the holes with great precision", "the movements are machine-like", "the reaction time is very high", "unnatural lateral movement while filling the holes", or "tries to needlessly fill regions without holes".

Hence, we conclude that the proposed method builds an agent with a human-like behavior by combining the goal-oriented learning scheme of RL and the tendency of the human expert's behavior.

4.3 Torcs

4.3.1 Experimental Setting

The Apple and Gopher game being cases of discrete action space environment, we sought to verify the effectiveness of the proposed method in a continuous action space case. Additionally, considering the potential real-world applicability and social impact of the proposed method, the rise in popularity of the autonomous driving task, as well as the need for autonomous agents to be able to interact with other human agents, we opted for the Torcs Racing Car Simulator [15], which is a well-known environment that is used for autonomous driving AI research (see [36], [37], as the representative task for the continuous action space case). A screenshot of the game is shown in Fig. 7. With respect to the experiment, we used the Gym Torcs environment [38] as a basis.

Fig. 7 A screenshot of Torcs.

The agent observations consisted of 65 features including LIDAR-like sensors keeping track of the distance between the car and edges of the track or the opponents, current speed and acceleration with a bi-dimensional continuous action space to control the steering and throttling of the car while using the raced distance as the reward function. This format allows for more precision when used as state representation for the agent, when compared to 2D pictures of the road for example. As a result, the potential noise and error caused by feature extractors can be virtually eliminated. This allows us to objectively train the different agents as well as evaluate their performance.

Furthermore, we upgraded the racing car simulator Torcs to simulate a situation where the human decision factor would stand out more. This was performed by generating obstacles in the form of stationary bots and adding player demonstration recording support, which was used to record 220 episodes of 60 s of gameplay, which corresponded to 792,000 transitions. The RL agent was based on the DDPG implementation in the OpenAI Baselines [33] and was adapted to the autonomous driving problem. This DDPG agent was first trained for 10^7 training timesteps by using the Adam [34] optimizer, learning rates of the critic and the actor components set to 10^{-4} and 10^{-3}, and a replay memory of size 10^6, before being used to generate the necessary trajectories needed to train the proposed hybrid model.

Table 3 Results of Torcs.

	Game Score				Sensitivity Test
Agent	**Max**	**Min**	**Average**	**Std.**	**Identified as Human (out of 52)**
Human	696.7	588.6	637.2	31.1	26
DDPG (RL)	823.4	818.8	**819.6**	0.5	16
GAIL (IL)	608.8	23.4	527.3	72.4	<u>27</u>
Proposed hybrid agent (RL+IL)	817.8	107.4	<u>661.2</u>	179.2	**32**

The best and second best results are emphasized by bold fonts and underlines, respectively.
The evaluated hybrid model corresponds to a trade-off coefficient of $\alpha = 0.5$

A strict IL part consisted in imitating a human expert based on recorded data of the latter while driving in the same environment configuration as the RL experiment via the OpenAI [33] implementation of the GAIL method and its default hyperparameter values except for the training timestep limit that was set to $5 \cdot 10^6$, empirically determined to allow convergence. The proposed hybrid agent based on the GAIL implementation with the discriminator loss modified based on Eq. (1) as described in Sect. 3.2 was also trained with the same environment configuration. The GAIL-based human imitation agent and hybrid agent were trained with the discriminator learning rate set to 10^{-3} and a KL constraint of 10^{-2} via the Adam optimizer [34] over $5 \cdot 10^6$ and $7.5 \cdot 10^6$ training timesteps, respectively. Furthermore, we explored the impact of the trade-off coefficient via experimenting with several values to determine the most appropriate value for the sensitivity test, following which we conducted a sensitivity test in a double-blind fashion to evaluate the human-likeliness of each reference method as well as the proposed one. Similarly to the Apple and Gopher games, the sensitivity test was also preceded by a trial phase, during which each participant got to know the rule of the game, as well as familiarize himself with the controls and game mechanics. In the Torcs Racing Car Simulator case, each participant was presented with two 30 seconds videos of each model playing the game and requested to label it as either a human or a machine, as well as providing comments which motivated such decision.

4.3.2 Results and Discussion

Finally, to evaluate the performance of a given agent in the Torcs Racing Car Simulator, we compute the average distance traveled by the said agent during a time frame of 5 seconds, to gauge the effective progress of the agents across the racing track. Also, the score for a time frame is set to 0 if the agent crashes or exits the racing track. We then investigated the impact of the trade-off coefficient on the hybrid agent. For $\alpha = 0.0$, the hybrid agent is equivalent to the default imitation of the human expert with the GAIL method. The hybrid agent's score then rises progressively with $\alpha \to 0.5$, thereby demonstrating a gain in performance obtained from the DDPG agent's demonstrations. From $\alpha = 0.5$ onward, the performance of the hybrid agent comes closer to that of the RL agent, but with moderate results for extreme values of α. Specifically, the deterministic nature of the DDPG agent's trajectories caused the hybrid agent to overfit the DDPG agent behavior, and thus it performed poorly during the testing phase.

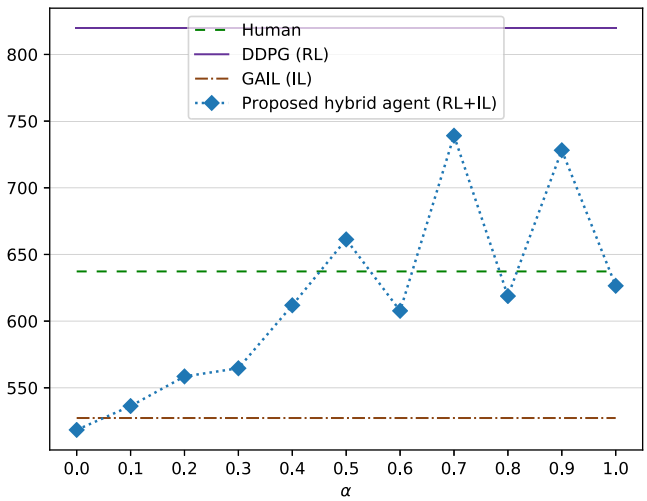

Fig. 8 Effect of the trade-off coefficient α on the proposed hybrid agent's score for Torcs. The average scores of the other agents are also provided for reference purposes.

In contrast to the Apple and the Gopher game, the performance gap between the RL agent and the human expert in this context was relatively lower. Moreover, the constraint of the "steering" and "acceleration" action to the $[-1, 1]$ range, as well as their continuous nature, resulted in easier to mix demonstrations for the hybridization process. A hybrid agent that performs better than the human expert can thus be obtained with smaller values of the trade-off coefficient α. Consequently, the hybrid agent corresponding to $\alpha = 0.5$ was used as the representative agent in the following experiments.

The first evaluation phase consisted in measuring the performance of the proposed model and comparing it relative to the human, GAIL human imitation, and DDPG agent respectively, as listed in Table 3.

The GAIL method can imitate expert trajectories generated by RL policies (results of [31]) or the deterministic bots coming with the Torcs Racing Car Simulator. However, it was less effective when applied on a human expert's trajectories and achieved approximately half the score of the latter at best. We hypothesized that this was because the human expert's policy was relatively more complex and diffi-

cult to approximate with standard neural network structures.

Furthermore, the proposed hybrid agent captured and demonstrated specific behaviors of the human expert and RL agent, namely a higher speed than the imitated human expert and deceleration during turns respectively. This was supported by the recurring comments collected from the participants of the sensitivity test, such as "the player decelerates before taking a turn or passing opponents", or "maintains a relatively constant speed overall, but with acceleration when the path follows a straight line". Similar comments were also given by the sensitivity test participants when characterizing the human expert demonstrations. Additionally, it also completed full laps like the human expert and DDPG agent.

The DDPG agent was least likely to be recognized as a human due to its high-performance behavior with comments such as "drives really fast" or "take turns at high speed". The other three agents were recognized as a human at a similar level. We thus concluded that the hybrid model was at least more likely to be recognized as a human than the DDPG agent, while still achieved a higher performance than the human agent and its imitation.

5. Conclusion

The study proposed a method to build a hybrid agent that behaves in a human-like fashion imitated from a human expert while retaining some of the high performance displayed by pure reinforcement learning agents with the ultimate aim of obtaining the best of both worlds. We based our method on state-of-the-art reinforcement and imitation learning algorithms and proposed two variants for discrete and continuous action spaces, respectively.

We applied the aforementioned method to an original game and a game of the Atari57 suite for the discrete action space case and the Torcs Racing Car Simulator for the continuous action space case. We evaluated its performance before evaluating its human-likeliness via a sensitivity test. While human-acceptable agents can be considered as the goal of this study, we used the human-likeliness of an agent as an alternative, albeit narrower measure for a human-acceptable behavior. The rationale is that an agent behaving in a way deemed human-like is also likely to be deemed as acceptable by other human players, when used as game AI for example. The "human-acceptable" criterion, however, would require a definition from scratch for each task, and its evaluation correspondingly difficult.

The agents trained using the proposed method successfully exhibited behaviors borrowed from both experts, namely an increase of performance following the reinforcement learning agent and a human-like behavior following the human counterpart. More specifically, the performance of the hybrid agents surpassed that of the human expert in each environment experimented on. A thorough analysis of comments collected from the participants of the double-blind sensitivity test further showcased the similarity in behavior of both the proposed agents and the corresponding human expert. In the light of the conducted experiments, the sensitivity evaluation approach adopted in this study would benefit from a larger population sample, and preferably formed with participants that are familiar enough with the games and simulations that were used to train the different agents.

Acknowledgments

This work was supported by JSPS KAKENHI (19H04172), by JST-Mirai Program Grant Number JPMJMI18B4, Japan, and by EQUOS RESEARCH Co., Ltd. The double-blind sensitivity test conducted in this study was organized and held by EQUOS RESEARCH Co., and the participants were selected from its employees. The first author would like to thank the Japanese Ministry of Education, Culture, Sports, Science, and Technology for its sponsoring via a scholarship from the year 2017 to 2020.

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. 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, 2017.
[2] A.E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," Electronic Imaging, vol.2017, no.19, pp.70–76, 2017.
[3] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," CoRR, abs/1610.03295, 2016.
[4] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," Proceedings of the International Conference on Robotics and Automation (IEEE ICRA), pp.2034–2039, 2018.
[5] B. Vikas, "Deep reinforcement learning approach to autonomous navigation," https://github.com/bhanuvikasr/Deep-RL-TORCS, 2017.
[6] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," Nature, vol.518, no.7540, pp.529–533, 2015.
[7] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," Proc. Int. Conf. Mach. Learn. (ICML), vol.48, pp.1928–1937, 2016.
[8] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," Proc. Int. Conf. Artif. Intell. Stat. (AISTATS), vol.15, pp.627–635, 2011.
[9] J. Ortega, N. Shaker, J. Togelius, and G.N. Yannakakis, "Imitating human playing styles in super mario bros," Entertainment Computing, vol.4, no.2, pp.93–104, 2013.
[10] K.-W. Chang et al., "Learning to search better than your teacher," Proc. Int. Conf. Mach. Learn. (ICML), vol.37, pp.2058–2066, 2015.
[11] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," Proceedings of International Conference on Robotics and Automation (IEEE ICRA), pp.6292–6299, 2018.
[12] T. Hester et al., "Learning from demonstrations for real world reinforcement learning," CoRR, abs/1704.03732, 2017.
[13] H.M. Le et al., "Hierarchical imitation and reinforcement learning,"

CoRR, abs/1803.00590, 2018.
[14] M.G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," Journal of Artificial Intelligence Research, vol.47, pp.253–279, 2013.
[15] B. Wymann, C. Dimitrakakis, A.D. Sumner, E. Espié, C. Guionneau, and R. Coulom, "Torcs: The open racing car simulator," 2013.
[16] R.F.J. Dossa, X. Lian, H. Nomoto, T. Matsubara, and K. Uehara, "A human-like agent based on a hybrid of reinforcement and imitation learning," Proceedings of the International Joint Conference on Neural Networks (IJCNN), 2019.
[17] H. Daumé III, J. Langford, and S. Ross, "Efficient programmable learning to search," CoRR, abs/1406.1837, 2014.
[18] J. Rao Doppa, A. Fern, and P. Tadepalli, "Hc-search: A learning framework for search-based structured prediction," The Journal of Artificial Intelligence Research (JAIR), vol.50, pp.369–407, 2014.
[19] S. Ross and J.A. Bagnell, "Reinforcement and imitation learning via interactive no-regret learning," CoRR, abs/1406.5979, 2014.
[20] Z. Wang et al., "Dueling network architectures for deep reinforcement learning," Proc. Int. Conf. Mach. Learn. (ICML), vol.48, pp.1995–2003, 2016.
[21] B. Balaguer and S. Carpin, "Combining imitation and reinforcement learning to fold deformable planar objects," Proceedings of the International Conference on Intelligent Robots and Systems (IEEE/RSJ IROS), pp.1405–1412, 2011.
[22] W. Sun, J.A. Bagnell, and B. Boots, "Truncated horizon policy search: Combining reinforcement learning & imitation learning," International Conference on Learning Representations (ICLR), 2018.
[23] A.Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," Proc. Int. Conf. Mach. Learn. (ICML), pp.278–287, 1999.
[24] S. Hecker, D. Dai, and L. van Gool, "Learning accurate, comfortable and human-like driving," abs/1903.10995, 2019.
[25] R.S. Sutton et al., Reinforcement Learning: An Introduction. MIT Press, 1998.
[26] T.P. Lillicrap et al., "Continuous control with deep reinforcement learning," CoRR, abs/1509.02971, 2015.
[27] D. Silver et al., "Deterministic policy gradient algorithms," Proc. Int. Conf. Mach. Learn. (ICML), vol.32, pp.I–387–I–395, 2014.
[28] M. Plappert et al., "Parameter space noise for exploration," CoRR, abs/1706.01905, 2017.
[29] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," arXiv preprint, arXiv:1503.02531, 2015.
[30] A.A. Rusu et al., "Policy distillation," CoRR, abs/1511.06295, 2015.
[31] J. Ho and S. Ermon, "Generative adversarial imitation learning," CoRR, abs/1606.03476, 2016.
[32] J. Schulman et al., "Trust region policy optimization," arXiv e-prints, arXiv:1502.05477, 2015.
[33] P. Dhariwal et al., "Openai baselines," https://github.com/openai/baselines, 2017.
[34] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," CoRR, 2014.
[35] N. Srivastava et al., "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, pp.1929–1958, 2014.
[36] B. Lau, "Using keras and deep deterministic policy gradient to play torcs," https://yanpanlau.github.io/2016/10/11/Torcs-Keras.html, 2016.
[37] Y. You, "Torcs for reinforcement learning," https://github.com/YurongYou/rlTORCS
[38] N. Yoshida, "Gym torcs," https://github.com/ugo-nama-kun/gym-torcs, 2016.

Rousslan F. J. Dossa received his B.S. in Computer system analysis and programming at the National School of Applied Economics and Management, Republic of Benin, in 2016. He then received his M.D. in Computational Science from the Graduate School of System Informatics at Kobe University in Japan in 2020. He is currently pursuing a Ph.D. degree at the Graduate School of System Informatics, Kobe University in Japan. His current research interests span across engineering, optimization theory, neuroscience, and deep learning, with a particular focus on deep reinforcement learning and self-driving cars, unsupervised learning, neural network architecture, and evolutionary computation.

Xinyu Lian received an M.S. degree from the Graduate School of System Informatics, Kobe University in 2019. Her research interests include deep reinforcement learning, especially its application to the development of human-like AI agents.

Hirokazu Nomoto received his B.E. degree in Faculty of Engineering from Tokyo University of Agriculture and Technology, Tokyo, Japan, in 1989. He is now working at EQUOS RESEARCH Co., Ltd. in the Tokyo head office laboratory, Tokyo, Japan. He is engaged in research and development of automotive-related technology.

Takashi Matsubara received his B.E., M.E., and Ph.D. in engineering degrees from Osaka University, Osaka, Japan, in 2011, 2013, and 2015, respectively. From 2015 to 2020, he was an Assistant Professor at the Graduate School of System Informatics, Kobe University, Hyogo, Japan. He is currently an Associate Professor at the Graduate School of Engineering Science, Osaka University, Osaka, Japan. His research interests are in deep learning for Bayesian modeling and dynamical systems.

Kuniaki Uehara received his B.E., M.E., and D.E. degrees in information and computer sciences from Osaka University, Osaka, Japan, in 1978, 1980 and 1984, respectively. From 1984 to 1990, he was with the Institute of Scientific and Industrial Research, Osaka University as an Assistant Professor. From 1990 to 1997, he was an Associate Professor with the Department of Computer and Systems Engineering at Kobe University. From 1997 to 2002, he was a Professor with the Research Center for Urban Safety and Security of Kobe University. From 2002 to 2020, he was a Professor with the Graduate School of System Informatics of Kobe University. Currently, he is a Professor with Faculty of Business Administration of Osaka Gakuin University.