Learning with Generated Teammates to Achieve Type-Free Ad-Hoc Teamwork

Dong Xing$^{1,2}$, Qianhui Liu$^{1,2}$, Qian Zheng$^3$ and Gang Pan$^{1,2}$*

$^1$College of Computer Science and Technology, Zhejiang University, Hangzhou, China
$^2$Zhejiang Lab, Hangzhou, China
$^3$School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore

{dongxing, qianhuiliu}@zju.edu.cn, zhengqian@ntu.edu.sg, gpan@zju.edu.cn

Abstract

In ad-hoc teamwork, an agent is required to cooperate with unknown teammates without prior coordination. To swiftly adapt to an unknown teammate, most works adopt a type-based approach, which pre-trains the agent with a set of pre-prepared teammate types, then associates the unknown teammate with a particular type. Typically, these types are collected manually. This hampers previous works by both the availability and diversity of types they manage to obtain. To eliminate these limitations, this work addresses to achieve ad-hoc teamwork in a type-free approach. Specifically, we propose the model of Entropy-regularized Deep Recurrent Q-Network (EDRQN) to generate teammates automatically, meanwhile utilize them to pre-train our agent. These teammates are obtained from scratch and are designed to perform the task with various behaviors, therefore their availability and diversity are both ensured. We evaluate our model on several benchmark domains of ad-hoc teamwork. The result shows that even if our model has no access to any pre-prepared teammate types, it still achieves significant performance.

1 Introduction

Cooperating with unknown teammates without prior coordination is a well-recognized challenge in multi-agent systems, known as ad-hoc teamwork [Stone et al., 2010]. For example, consider a scenario where several agents are assembled to participate in an impromptu soccer game. Due to limited time, these agents cannot coordinate in advance to determine each other’s division of work. However, only when they cooperate effectively can the task be performed with high quality. In such scenarios, the agent is expected to cooperate with some previously unknown teammates, but a prior coordination protocol among them does not exist. As agents proliferate in the real world and their functions become more specialized, there will be an increasing number of scenarios that require the agent to participate in such ad-hoc teamwork.

One critical requirement for ad-hoc teamwork is to let the agent swiftly adapt to an unknown teammate. To meet this requirement, most existing works adopt a type-based approach [Albrecht and Stone, 2018]. This approach pre-trains the agent with a set of pre-prepared teammate types, then associates the unknown teammate with a particular type. In this way, the agent can reuse the strategies learned from past interactions with these available types. Typically, these teammate types are collected manually [Barrett et al., 2017; Ravula et al., 2019]. This hampers most type-based works in two aspects. First, a set of pre-prepared teammate types is not always available. Since these teammates are task-specific, it is likely that we are not able to obtain a set of qualified teammate types on a novel task. Second, even with some types available, previous works provide little guarantees on the diversity of their behaviors. If an unknown teammate behaves differently from any of our pre-prepared types, the effect of those pre-trained strategies will be reduced.

To eliminate the aforementioned limitations, this work addresses to achieve ad-hoc teamwork in a type-free approach (we use type-free to describe methods that do not require users to provide a pre-prepared set of teammate types). Specifically, we propose the model of Entropy-regularized Deep Recurrent Q-Network (EDRQN) (Section 4.1) to automatically generate teammates from scratch. These teammates are designed to perform the task with various behaviors, so that our agent can learn to cooperate with a variety of them. Compared with the type-based approach, the teammates generated by EDRQN share the following two properties:

1. Availability (Section 4.2): our model generates teammates from scratch and this process does not require any domain expertise from the user. This ensures that our teammates are available to a wide range of tasks, which enriches our application on ad-hoc teamwork.

2. Diversity (Section 4.3): we cover the diversified behaviors of teammates with an episodic-wise entropy regularizer. This allows our agent to cooperate with an infinite number of potential teammates, rather than being limited to a finite set of pre-prepared types.

The core contribution of this work is a type-free model to generate teammates that are available and diverse on a wide range of ad-hoc teamwork tasks. This frees us from the burden of manually collecting a pre-prepared set of teammates,
which are required for most type-based works. We demonstrate the performance of our model on several benchmark domains that are frequently adopted among ad-hoc teamwork researches, including the pursuit domain and the half field offense domain.

2 Related Work

This section discusses related works in the field of ad-hoc teamwork. According to the dependency of a pre-prepared set of teammate types, we divide relevant researches into two categories: type-based ad-hoc teamwork and type-free one.

2.1 Type-Based Ad-Hoc Teamwork

Type-based approach is a widely adopted solution for ad-hoc teamwork [Albrecht and Stone, 2018] since it can largely utilize existing models to adapt to unknown teammates. A critical step of this approach is to determine the unknown teammate’s type with limited observations. To solve this, several works [Barrett et al., 2011; Barrett et al., 2017] inferred the teammate’s type with a posterior distribution and updated it by Bayes’s law. In [Ravula et al., 2019], a change point detection algorithm was proposed to detect the potential switching of teammate types. Recently, [Chen et al., 2020] applied attention mechanism to dynamically fuse the candidate types. These methods all require the users to collect a prior set of teammate types, of which the availability and diversity are hard to guarantee. Instead, we generate these teammates from scratch and let them perform the task with various styles, so our model is not constrained by these limitations.

The workload of collecting teammate types has been alleviated in some works. For example, in [Albrecht et al., 2015], the set of teammate types were generated with several artificially designed rules. In [Mirskey et al., 2020], a cooperative tool fetching task was proposed, and the authors represented the teammate’s type with the tool’s location, which determined the teammate’s strategy. However, these methods are deeply tied to their domains, making them hard to apply to a new task. Instead, our model’s process to generate teammates is fully task-independent.

2.2 Type-Free Ad-Hoc Teamwork

Although being rare, there exist some works that address ad-hoc teamwork in a type-free manner. In [Wu et al., 2011], the authors predicted the teammate’s future action with its recent plays and then performed online planning based on the prediction. However, the planning requires a complete model of the environment, which can be unavailable if the task is complex. In [Canaan et al., 2019], the authors proposed a framework to generate players for a card game named Hanabi. However, this framework requires the user’s domain expertise to locate the teammate in a behavior space [Mouret and Clune, 2015], which can be hard to transfer to tasks other than Hanabi. In [Hu et al., 2020], a problem termed zero-shot coordination was proposed, which requires several agents (who do not know each other) to cooperate within limited interactions. The goal of [Hu et al., 2020] is closely related to that of ad-hoc teamwork. However, they assume that all the agents are optimized for the zero-shot setting, which does not apply to our scenario since the unknown teammate’s strategy cannot be manipulated in ad-hoc teamwork. To our knowledge, on ad-hoc teamwork, a universal type-free model to generate teammates that are both available and diverse is still missing, and we are the first attempt to fill this gap.

3 Preliminaries

This section provides preliminaries of our work. We first review the framework of Partially Observable Stochastic Game (POSG) [Hansen et al., 2004], which models the interaction of ad-hoc teamwork with partial observation. Then, we introduce the deep Q-learning and its variation deep recurrent Q-learning, which together form the foundation of our model.

3.1 Partially Observable Stochastic Game

The Partially Observable Stochastic Game (POSG) is a suitable framework to model the interaction of ad-hoc teamwork, since in real-world an agent can rarely observe a full state from the environment, and the cooperation requires each agent to interact in a stochastic game. Specifically, a POSG is defined as a tuple \( \langle I, S, \{ A_i \}, \{ O^i \}, P, R, \gamma \rangle \), where:

- \( I \) is the set of agents, denoted by \( i \in \{ 1, 2, \ldots, n \} \).
- \( S \) is the set of state \( s \) for all the agents.
- \( A^i \) is the set of action \( a^i \) for agent \( i \) (with a capacity of \( N \)), and \( A = \times_{i \in I} A^i \) is the set of joint action \( a \).
- \( O^i \) is the set of observation \( o^i \) for agent \( i \), and \( O = \times_{i \in I} O^i \) is the set of joint observation \( o \).
- \( P \) is the transition function, and \( P(s^t, o^t|s_t, a_t) \) is the probability of transiting to state \( s^t \) and receiving joint observation \( o^t \) when taking joint action \( a_t \) at state \( s_t \).
- \( R : S \times A \rightarrow \mathbb{R} \) is the reward function, which is shared by all agents since we focus on cooperative tasks.
- \( \gamma \in (0, 1] \) is the discount factor for the future reward.

At every timestep \( t \), each agent \( i \in I \) receives a partial observation \( o^i_t \in O^i \) from the environment, based on which the agent \( i \) determines an action \( a^i_t \in A^i_t \). The environment generates a reward \( r^i_t = R(s^t, a^i_t) \) for all agents according to the current state \( s_t \in S \) and the joint action \( a_t \in A \). Then, it transits to the next state \( s^t_t \in S \) and outputs a joint observation \( o^t_t \in O \) with respect to the transition function \( P \). The goal of each agent is to learn a policy \( \pi^i \) that maximizes the team’s expected cumulative reward \( \mathbb{E} \left[ \sum_t \gamma^t r^i_t \right] \).

3.2 Deep Q-Learning

Q-Learning [Watkins and Dayan, 1992] is a model-free off-policy algorithm to estimate the agent \( i \)'s expected cumulative reward \( Q^i(s_t, a^i_t) \) when performing action \( a^i_t \) at state \( s_t \). Specifically, for each state-action pair, the Q-value is updated towards the received reward \( r_t \) plus the maximal Q-value over all actions \( a^i_t \) at the following state \( s^t_t \):

\[
Q^i(s_t, a^i_t) \leftarrow r_t + \gamma \mathbb{E}_{a^i_t'} \left[ \max_{a^i_t'} Q^i(s^t_t, a^i_t') \right]
\]

In a complex scenario, the state space can be infinitely large, and it becomes infeasible to store the estimated Q-values for
all state-action pairs [Yang et al., 2020]. Instead, a Deep Q-
Network (DQN) is proposed in [Mnih et al., 2015] to approx-
imate the Q-value. This network is parameterized with $\theta$, and
the loss function is defined by minimizing the difference be-
tween the current Q-value and its target:

$$
\min L(\theta) = \mathbb{E} \left[ \left( r_t + \gamma \max_{a'_t} Q_\theta \left( s_{t'}, a'_t \right) - Q_\theta \left( s_t, a_t \right) \right)^2 \right]
$$

### 3.3 Deep Recurrent Q-Learning

The output of DQN is fully conditioned on the current state $s_t$, whose performance could be affected if the agent only
obtains a partial observation from the environment [Yang et al., 2018; Meng et al., 2021]. Instead, Deep Recurrent Q-
Network (DRQN) [Hausknecht and Stone, 2015] makes use
of past observations to compensate for the unobserved in-
formation of the current state. Briefly, DRQN is a model
that replaces the decision layer of DQN with a recurrent net-
work [Hochreiter and Schmidhuber, 1997]. In DRQN, the Q-
value $Q(o_i, a_i)$ represents the expected cumulative rewards
for action $a_i$ given the observation $o_i$ (instead of the state $s_i$). Besides, this value is also conditioned on past observations $o_{0:t-1}$, so that even if the information at timestep $t$ is missing, the agent can still infer a proper action with past observations.

### 4 Method

This section provides the details of our method. We first de-
scribe our proposed EDRQN model in Section 4.1. Then, we
discuss the availability and diversity of teammates generated
by our model in Section 4.2 and 4.3 respectively.

#### 4.1 The EDRQN model

**An Episodic-wise Objective**

Recall that our goal is to generate teammates who perform the
task with diversified behaviors, so that our agent can pre-train
with a variety of them. Regarding the teammate’s diversity, a
direct metric is the entropy of its policy. Thus, based on the
framework of POSG, we set the teammate’s objective as:

$$
\max_{\pi_i} \mathbb{E} \left[ \sum_{t} \gamma^t r_t \right] \quad \text{s.t.} \quad \mathcal{H} (\pi_i) = \bar{\mathcal{H}}_j, \ \forall t
$$

where $\bar{\mathcal{H}}_j$ represents the target entropy of agent $i$’s policy at $j$-th episode (we use agent $i$ to formally denote the
teammate). This target entropy changes on different episodes and
each value represents a specific type of teammate. Therefore,
our objective is made episodic-wise. Restricting the entropy
of an agent’s policy to a fixed value derives from maximum
entropy reinforcement learning (MaxEnt RL), which has been
previously applied to promote an agent’s multi-mode behav-
iors [Fox et al., 2016; Haarnoja et al., 2018]. However, to our
knowledge, no previous work has practiced to work with an
episodic-wise objective as we do. When considering ad-hoc
teamwork, a changing target entropy is essential, because it
allows our model to cover teammates with extremely different
styles of behaviors. For example, a nearly-zero $\bar{\mathcal{H}}_j$ corre-
sponds to an (almost) deterministic policy, which represents
a prudent teammate who only picks the optimal action. In
contrast, a large $\bar{\mathcal{H}}_j$ corresponds to a stochastic policy, which
represents a teammate who chooses actions more casually. It
is clear that these teammates, with completely different styles
of behavior, cannot be represented with a common target en-
try. Thus, an episodic-wise objective is necessary.

**Two Piled Networks**

By using Q-values to represent the discounted cumulative re-
wards and then expressing Eq. 1 with its dual form, we can re-
formulate our objective as:

$$
\min_{\alpha_{i}^t} \max_{\pi_i} \mathbb{E} \left[ Q_{\theta} \left( o_{i}^t, a_{i}^t \right) \right] + \alpha_{i}^t \left( \mathcal{H} (\pi_i) - \bar{\mathcal{H}}_j \right)
$$

where $\alpha_{i}^t$ is a Lagrange multiplier, and the entropy of policy
$\pi_i^t$ is formed as a regularizer. Given $\alpha_{i}^t$, it can be proved that
the optimal policy is the following Boltzmann policy:

$$
\pi_i^{t} (a_i | o_{i}^t, \alpha_{i}^t) = \softmax_{a_i} \left( Q_{\theta} \left( o_{i}^t, \cdot \right) / \alpha_{i}^t \right)
$$

where we can notice that $\alpha_{i}^t$ tunes agent $i$’s final policy (and therefore its entropy) given Q-values. By inspecting the rel-
ationships among $o_{i}^t$, $Q_{\theta}$, $\alpha_{i}^t$ and $\pi_i^t$, we now introduce two
piled networks. The first network (marked with orange back-
grounds in Figure 1) represents the Q network, which takes
the current observation $o_{i}^t$ as input and then generates Q-
values for each action $\{a_1, \ldots, a_N\}$. The second network
(marked with green backgrounds in Figure 1) represents the
$\alpha_{i}^t$ network, which takes the Q-values as input and then out-
puts a proper result to tune the policy. These two networks are
both implemented with Gated Recurrent Units (GRU) [Cho et
al., 2014] to make better use of previous observations to com-
pensate for the unobserved information of the current state.

**A Closed-loop Target Entropy**

One concern for the episodic-wise target entropy is that the
objective of Eq. 1 is erratic. With a same input, the objective
can be different for even two succeeding episodes, depending
on the gap between the sampled $\bar{\mathcal{H}}_j$ and $\bar{\mathcal{H}}_j^t$. As a result, the
model will be optimized to a teammate that averages all the
sampled targets, which is against our intention. To solve this,
we let the sampled $\tilde{H}_j^i$ be a part of input for our two networks (denoted with blue arrows in Figure 1), so that these networks are aware of what objective they are dealing with at current episode. This makes the target entropy closed-loop, since it simultaneously appears in the input and output (the objective) of our model during training.

Albeit its simplicity, there are several benefits to make the target entropy closed-loop. First, conditioned on a given $\tilde{H}_j^i$, our objective now becomes unique, which helps stabilize the training. Second, the varying target simulates an infinite variety of teammate types. If teammates with similar target entropies also behave alike, this setting can cover teammates unseen before with the help of model’s generalization. These benefits promote the diversity of our generated teammates.

The Optimization
Similar to [Haarnoja et al., 2018], we optimize our model with the dual gradient descent, which alternates between the optimization of $Q^i_\theta$ and $\alpha^i_j$. Following MaxEnt RL, the optimization of $Q^i_\theta$ is set as:

$$Q^i_\theta(\alpha^i_j, \alpha^i_0) \leftarrow r^i_t + \gamma \mathbb{E}_{o^i_t} \left[ \alpha^i_j \log \sum_\alpha^i \frac{Q^i_\theta(o^i_0, \alpha^i_0)}{\alpha^i} \right]$$

Meanwhile, we set the objective of $\alpha^i_t$ as:

$$\alpha^i_t = \arg \min_{\alpha^i_t} \mathbb{E}_{o^i_t} \left[ ||H(\pi^i(o^i_t, \alpha^i_t)) - \tilde{H}_j^i||^2_2 \right]$$

where the expectation is taken over past interactions sampled from a replay buffer. The training is conducted through self-play. At the beginning of a new episode, each agent samples an independent target entropy to simulate a specific type of player. Then, these agents interact in the task and simultaneously update their models with past experiences. Please note that each agent does not rely on the target entropy of other agents. This ensures that our agent can work with a teammate whose target entropy is agnostic during the evaluation.

4.2 Availability of Teammates
Recall that a major limitation for most type-based works is that they are based on a pre-prepared set of teammate types. These teammate types are task-specific and typically require the user to collect them manually. However, it is likely that we cannot obtain such a pre-prepared set of qualified teammate types on a novel task, especially when the task we are dealing with is complex. With EDRQN, this limitation is eliminated. From Section 4.1 we can verify that with our model, an infinite number of diversified teammates can be generated from scratch through self-play. This process is conducted automatically, with no requirement for the user’s domain expertise. These properties ensure that our generated teammates are available on a wide range of ad-hoc teamwork tasks, which free us from the burden of manually collecting a pre-prepared set of teammate types.

4.3 Diversity of Teammates
Another uncertain factor for most type-based works is that they provide little guarantees on the diversity of their collected teammate types. This affects their application in practice, because if an unknown teammate behaves differently from all the pre-prepared types, the effect of all the agent’s pre-trained strategies will be reduced. With EDRQN, the diversity of our generated teammates can be covered in two aspects. First, our episodic-wise objective in Eq. 1 enables the generated teammate to simulate an infinite variety of behaviors. This claim is further supported in Section 5.2, where we can quantitatively measure an agent’s diversity by its performance on a specially designed task. Second, throughout the experiment, our teammate adopts a stochastic Boltzmann policy (Eq. 3), which also promotes its diversity by picking not only the optimal action, but also those sub-optimal ones.

5 Experiment
This section provides the experimental studies. We first introduce the specifications of our benchmark domains, including the pursuit domain and the half field offense (HFO) domain. Then, we analyze our model’s performance on each domain.

5.1 Domain Specifications
Pursuit
The pursuit domain [Benda, 1986] (Figure 2a) has been frequently adopted in the research of ad-hoc teamwork [Barrett et al., 2011; Ravula et al., 2019], since this task cannot be completed by a single agent, regardless of its ability. The details of this domain vary in different works, but they all revolve around a group of predators capturing one or more moving preys with minimum timesteps. Typically, the world is formed with a grid of cells. At each timestep, both the predators and preys choose to either move to a neighboring cell or stay in their current position. A prey is captured if it is surrounded by a certain number of predators.

In our version of the pursuit domain, the world is a toroidal world of size $28 \times 28$. This means that if an agent moves off one end of the world, it comes back on the other end. There are 2 predators (marked with orange balls in Figure 2a) in our domain, and we are in control of one predator during the evaluation. Besides, there are 4 preys (marked with blue balls), and their ranges of movement are limited to 4 separate $4 \times 4$ sub-grids (marked with light blue background). This setting allows us to quantitatively measure the behavior diversity of the agent, since each predator can be featured by the distribution of different preys it captured. A prey does not move...
if it tries to run out of the boundary or a predator is at its neighboring cells. If two agents run into a same cell, the collision is solved randomly. To simulate a partial observation, each predator is limited to observe the locations of itself, its teammate and the nearest prey. The task is completed if two predators are next to a prey within a maximal step of 300. Otherwise, the task is considered failed.

**Half Field Offense**

The half field offense (HFO) domain [Hausknecht et al., 2016] (Figure 2b) originates from the RoboCup 2D soccer simulation league. This domain is challenging because it largely retains the complexity and uncertainty of the RoboCup 2D soccer simulation league. Besides, it requires high-level cooperation from the agents to score a goal, making it one of the most suitable domains to evaluate the ad-hoc teamwork [Barrett et al., 2017]. In this domain, a group of offensive agents try to score a goal against a group of defensive agents (including a goalie). The game ends when any of the following criteria is reached: 1) the offensive team scores a goal, 2) the defensive team captures the ball, 3) the ball is kicked out of bounds, or 4) the game lasts for 500 simulation steps (50 seconds).

Our setting in this domain follows the work of [Chen et al., 2020]. In this game, two offensive agents (marked with orange balls in Figure 2b) play against two defensive agents (marked with a purple ball for the goalie and a blue ball for the other defender), and we are in control of one offensive agent during evaluation. Throughout the experiment, the defensive team adopts the strategy provided by agent2d.

During evaluation, our teammate is sampled from 10 independent participants of the 2013 RoboCup 2D soccer simulation league. To simulate a partial observation, each agent is limited to observe the current positions of all agents and the ball. The remaining information (such as the direction and speed of each agent, the opening angle to the goal) should be inferred from the context. For each simulation step, the agent picks a high-level action from \{DRIBBLE, PASS, SHOOT\} if it is in control of the ball, otherwise it performs the default MOVE action. These actions are provided by the HFO codebase to decrease the complexity of agent’s action space.

### 5.2 Evaluation on the Pursuit Domain

For the pursuit domain, we choose the PLASTIC model proposed in [Barrett et al., 2017] to compare with our model, which is a state-of-the-art work on this domain. PLASTIC is a classic type-based model, whose implementation can be divided into 3 steps: 1) learn a policy for each collected teammate type, 2) update beliefs over all types to pick the closest one with respect to the unknown teammate’s behavior, and 3) select actions by the policy learned for the closest teammate type. We first train a set of 18 DQN agents via self-play, which are all able to perform the task with over 95% success rate. We choose DQN because it can be viewed as EDRQN with a zero target entropy, whose diversity is constrained and therefore forms an ordinary teammate that can be encountered in reality. Then, for the first step of PLASTIC, we sample a subset from these agents to simulate the teammate types that are manually collected by the user, and represent the learned policies for these teammates by those of their partners during the self-play. The remaining two steps of PLASTIC are performed by picking the optimal policy from our candidate set in hindsight, which forms an upperbound of PLASTIC (and many other type-based methods). Please note that this implementation is infeasible in practice, because it requires us to foresee the outcome of an unknown teammate’s cooperation with all our pre-prepared policies before the task begins. Nevertheless, it constitutes a challenging target for our model to compare with.

Our comparison result is illustrated in Figure 3. For each DQN agent $i \in [1, 18]$ (indexed at the bottom-right corner of each sub-figure), we let the remaining 16 agents (the ones other than $i$ and its self-play partner) to form the whole set of teammate types which the user can collect from. Since it is costly for a user to obtain a teammate type in a real-world scenario, we present the result of PLASTIC when the user is limited to collect teammate types with a capacity of 1, 2, 4 and 8 (denoted as $C_1$, $C_2$, $C_4$ and $C_8$, respectively). We also present the agent’s performance with its original partner as a baseline. It can be observed that of all the 18 un-
known teammates, our EDRQN outperforms in 13 of them, which is higher than the result of PLASTIC. Although the performance of PLASTIC is improved along with more collected types, this can be hard to achieve in practice because as the task becomes more complex, the difficulty of obtaining a qualified teammate type is also rising. Instead, our EDRQN does not require such a pre-prepared set of types.

We further examine the diversity of our generated teammates by inspecting the distributions of their captured preys. Figure 4 presents the heatmaps of preys captured by each pair of self-play agents, together with the entropies of their distributions. It can be observed that the entropies of our EDRQN teams are higher than DQN teams on average. By inspecting the distributions of captured preys, the results of EDRQN teams are even more, which suggests that their ways to perform the task are more diverse than DQN teams. This property enables our agent to cooperate with unknown teammates of various types more smoothly.

5.3 Evaluation on the HFO Domain

On HFO domain, apart from the PLASTIC model, we additionally choose the AATEAM model proposed in [Chen et al., 2020] for comparison. The AATEAM model is another type-based approach, and its major difference with the PLASTIC model is the way of picking the closest type. For AATEAM, an attention mechanism is designed to fuse the prior beliefs over all available teammate types. Nevertheless, these two models both require the user to provide a set of pre-prepared teammate types before the task begins. We follow the experimental setting of [Chen et al., 2020] and let the teams of agent2d, aut, axiom, gliders and yushan be the pre-prepared teammate types collected by the user, and let the teams of legend, ubc, utaustin, warthog and yunlu be the unknown teammates that require our agent to cooperate with in an ad-hoc teamwork setting. Note that this division only applies to the type-based approach (herein the PLASTIC model and the AATEAM model). From the perspective of EDRQN, all the teammates are unknown (including teams in the pre-prepared set) when we pre-train our model.

Our comparison result is presented in Table 1. The baseline shows the performance of agents playing with their original teammates. We can observe that among all the methods, our model’s performance is competitive. For teams in the pre-prepared set, our model is leading in 2 of the 5 teams, which are also 2 best scores among these teams. This is non-trivial, considering the fact that during the pre-training, all these teams are accessible to PLASTIC and AATEAM, but are unknown to our model. For teams not in the pre-prepared set, the task becomes more challenging for type-based methods. As evidence, the average performance of PLASTIC drops from 73.4% to 70.3%, and that of AATEAM drops from 76.5% to 73.2%. Nevertheless, EDRQN is not affected by this condition and leads in 4 of 5 teams, with an average scoring rate of 77.3%. These teams are developed by independent groups and therefore their strategies vary from each other. However, our agent is able to respond properly to most of them. This again shows that even if our model has no access to any pre-prepared teammate types, it can still achieve significant performance on complex tasks such as HFO.

6 Conclusion

This paper presents the proposed EDRQN model to achieve ad-hoc teamwork in a type-free approach. By pre-training agents with an episodic-wise objective through self-play, our EDRQN model is able to generate teammates that perform the task with various behaviors. This ensures that our teammates are available and diverse to a wide range of tasks, which are nontrivial to guarantee for most existing type-based works. We evaluate our model on two representative domains of ad-hoc teamwork. With detailed analyses, we show that even if our model has no access to any pre-prepared teammate types, it can still achieve significant performance. A natural extension of our work is to apply EDRQN on scenarios with more teammates. This can be challenging because the interactions are more complex now, and we leave it as a future work.

Acknowledgements

This work was supported by the Natural Science Foundation of China (No. 61925603), Zhejiang Lab and Ten Thousand Talent Program of Zhejiang Province (No. 2018R52039).
References


