ROMA: Multi-Agent Reinforcement Learning with Emergent Roles

Tonghan Wang  1  Heng Dong  1  Victor Lesser  2  Chongjie Zhang  1

Abstract

The role concept provides a useful tool to design and understand complex multi-agent systems, which allows agents with a similar role to share similar behaviors. However, existing role-based methods use prior domain knowledge and predefined role structures and behaviors. In contrast, multi-agent reinforcement learning (MARL) provides flexibility and adaptability, but less efficiency in complex tasks. In this paper, we synergize these two paradigms and propose a role-oriented MARL framework (ROMA). In this framework, roles are emergent, and agents with similar roles tend to share their learning and to be specialized on certain sub-tasks. To this end, we construct a stochastic role embedding space by introducing two novel regularizers and conditioning individual policies on roles. Experiments show that our method can learn specialized, dynamic, and identifiable roles, which help our method push forward the state of the art on the StarCraft II micromanagement benchmark. Demonstrative videos are available at https://sites.google.com/view/romarl/.

1. Introduction

Many real-world systems can be modeled as multi-agent systems (MAS), such as autonomous vehicle teams (Cao et al., 2012), intelligent warehouse systems (Nowé et al., 2012), and sensor networks (Zhang & Lesser, 2011). Cooperative multi-agent reinforcement learning (MARL) provides a promising approach to developing these systems, allowing agents to deal with uncertainty and adapt to the dynamics of an environment. In recent years, cooperative MARL has achieved prominent progress, and many deep methods have been proposed (Foerster et al., 2018; Sunehag et al., 2018; Rashid et al., 2018; Son et al., 2019; Vinyals et al., 2019; Wang et al., 2020b; Baker et al., 2020).

In order to achieve scalability, these deep MARL methods adopt a simple mechanism that all agents share and learn a decentralized value or policy network. However, such simple sharing is often not effective for many complex multi-agent tasks. For example, in Adam Smith’s Pin Factory, workers must complete up to eighteen different tasks to create one pin (Smith, 1937). In this case, it is a heavy burden for a single shared policy to represent and learn all required skills. On the other hand, it is also unnecessary for each agent to use a distinct policy network, which leads to high learning complexity because some agents often perform similar sub-tasks from time to time. The question is how we can give full play to agents’ specialization and dynamic sharing for improving learning efficiency.

A natural concept that comes to mind is the role. A role is a comprehensive pattern of behavior, often specialized in some tasks. Agents with similar roles will show similar behaviors, and thus can share their experiences to improve performance. The role theory has been widely studied in economics, sociology, and organization theory. Researchers have also introduced the concept of role into MAS (Becht et al., 1999; Stone & Veloso, 1999; Depke et al., 2001; Ferber et al., 2003; Odell et al., 2004; Bonjean et al., 2014; Lhaksmana et al., 2018). In these role-based frameworks, the complexity of agent design is reduced via task decomposition by defining roles associated with responsibilities.
We test our method on StarCraft II\(^1\) micromanagement environments (Vinyals et al., 2017; Samvelyan et al., 2019). Results show that our method significantly pushes forward the state of the art of MARL algorithms, by virtue of the adaptive policy sharing among agents with similar roles.  

Visualization of the role representations in both homogeneous and heterogeneous agent teams demonstrates that the learned roles can adapt automatically in dynamic environments, and that agents with similar responsibilities have similar roles. In addition, the emergence and evolution process of roles is shown, highlighting the connection between role-driven sub-task specialization and improvement of team dynamics. We consider partially observable settings and agent \(i\) only has access to an observation \(o_i \in \Omega\) drawn according to the observation function \(O(s, i)\). Each agent has a history \(\tau_i \in T \equiv (\Omega \times A)^*\). At each timestep, each agent \(i\) selects an action \(a_i \in A\), forming a joint action \(\alpha = A^n\), leading to next state \(s'\) according to the transition function \(P(s'|s, \alpha)\) and a shared reward \(r = R(s, \alpha)\) for each agent. The joint policy \(\pi\) induces a joint action-value function: 

\[
Q^\pi_{\text{tot}}(s, \alpha) = \sum_{\alpha_0=\alpha} \sum_{s_0:s} \sum_{t=0}^{\infty} \gamma^t r_t \equiv E_{s_0, s, a_0, a=\alpha, \pi}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0=s, a_0=\alpha, \pi].
\]

To effectively learn policies for agents, the paradigm of centralized training with decentralized execution (CTDE) (Foerster et al., 2016; 2018; Wang et al., 2020a) has recently attracted attention from deep MARL to deal with non-stationarity while learning decentralized policies. One of the promising ways to exploit the CTDE paradigm is value function decomposition (Sunehag et al., 2018; Rashid et al., 2018; Son et al., 2019; Wang et al., 2020b), which learns a decentralized utility function for each agent and uses a mixing network to combine these local utilities into a global action value. To achieve learning scalability, existing CTDE methods typically learn a shared local value or policy network for agents. However, this simple sharing mechanism is often not sufficient for learning complex tasks, where diverse responsibilities or skills are required to achieve goals. In this paper, we develop a novel role-based MARL framework to address this challenge. This framework achieves efficient shared learning while allowing agents to learn sufficiently diverse skills.

### 3. Method

In this section, we will present a novel role-oriented MARL framework (ROMA) that introduces the role concept into MARL and enables adaptive shared learning among agents. ROMA adopts the CTDE paradigm. As shown in Fig. 2, it learns local Q-value functions for agents, which are fed into a mixing network to compute a global TD loss for centralized training. During the execution, the mixing network will be removed, and each agent will act based on its local policy derived from its value function. Agents’ value functions or policies are dependent on their roles, each of which is responsible for performing similar automatically identified sub-tasks. To enable efficient and effective shared learning among agents with similar behaviors, ROMA will automatically learn roles that are:

i) **Dynamic**: An agent’s role can automatically adapt to the dynamics of the environment;  

ii) **Identifiable**: The role of an agent contains enough information about its behaviors;  

iii) **Specialized**: Agents with similar roles are expected to specialize in similar sub-tasks.

---

\(^1\)StarCraft II are trademarks of Blizzard EntertainmentTM.
ROMA: Role-Oriented Multi-Agent Reinforcement Learning

Formally, each agent $i$ has a local utility function (or an individual policy), whose parameters $\theta_i$ are conditioned on its role $\rho_i$. To learn roles with desired properties, we encode roles in a stochastic embedding space, and the role of agent $i$, $\rho_i$, is drawn from a multivariate Gaussian distribution $\mathcal{N}(\mu_{\rho_i}, \sigma_{\rho_i})$. To enable the dynamic property, ROMA conditions an agent’s role on its local observations, and uses a trainable neural network $f$ to learn the parameters of the Gaussian distribution of the role:

$$
(\mu_{\rho_i}, \sigma_{\rho_i}) = f(o_i; \theta_\rho),
$$

where $\theta_\rho$ are parameters of $f$. The sampled role $\rho_i$ is then fed into a hyper-network $g(\rho_i; \theta_h)$ parameterized by $\theta_h$ to generate the parameters for the individual policy, $\theta_i$. We call $f$ the role encoder and $g$ the role decoder. In the next two sub-sections, we will describe two regularizers for learning identifiable and specialized roles.

### 3.1. Identifiable Roles

Introducing latent role embedding and conditioning individual policies on this embedding does not automatically generate roles with desired properties. Intuitively, conditioning roles on local observations enables roles to be responsive to the changes in the environment. This design enables ROMA to be adaptive to dynamic environments but may cause roles to change quickly, making learning unstable. For addressing this problem, we expect roles to be temporally stable. To this end, we propose to learn roles that are identifiable by agents’ long term behaviors, which can be achieved by maximizing $I(\tau_i; \rho_i | o_i)$, the conditional mutual information between the individual trajectory and the role given the current observation.

However, estimating and maximizing mutual information is often intractable. Drawing inspiration from the literature of variational inference (Wainwright et al., 2008; Alemi et al., 2017), we introduce a variational posterior estimator of variational inference (Wainwright et al., 2008; Alemi et al., 2017), we introduce a variational posterior estimator of variational inference (Wainwright et al., 2008; Alemi et al., 2017), we introduce a variational posterior estimator of variational inference (Wainwright et al., 2008; Alemi et al., 2017), we introduce a variational posterior estimator of variational inference.

![Figure 2. Schematics of our approach. The role encoder generates a role embedding distribution, from which a role is sampled and serves as the input to the role decoder. The role decoder generates the parameters of the local utility network. Local utilities are fed into a mixing network to get an estimation of the global action value. We propose two learning objectives to learn specialized and identifiable roles. The framework can be trained in an end-to-end manner.](image)

where $D$ is a replay buffer, and $\text{KL}[\cdot || \cdot]$ is the KL divergence operator. The detailed derivation can be found in Appendix A.1.

### 3.2. Specialized Roles

The formulation so far does not promote sub-task specialization, which is the critical component to share learning and improve efficiency in multi-agent systems. Minimizing $\mathcal{L}_I$ enables roles to contain enough information about long-term behaviors but does not explicitly ensure agents with similar behaviors to have similar role embeddings.

For learning specialized roles, we define another role-learning regularizer. Intuitively, to encourage sub-task specialization, for any two agents, we expect that either they have similar roles or they have quite different behaviors. However, it is usually unclear which agents will have similar roles during the process of role emergence, and the similarity between behaviors is not straightforward to measure.

Since roles have enough information about the behaviors (achieved by minimizing $\mathcal{L}_I$), to encourage two agents $i$ and $j$ to have similar roles, we can maximize $I(\rho_i; \tau_j)$, the mutual information between the role of agent $i$ and the trajectory of agent $j$. However, we do not know which agents will have similar roles, and directly optimizing this objective for all pairs of agents will result in all agents having the same role, and, correspondingly, the same policy, which
will limit system performance. To settle this issue, we introduce a dissimilarity model \( d_\phi : T \times T \rightarrow \mathbb{R} \), a trainable neural network taking two trajectories as input, and seek to maximize \( I(\rho_i; \tau_j) + d_\phi(\tau_i, \tau_j) \) while minimizing the number of non-zero elements in the matrix \( D_\phi = (d_{ij}) \). Here, \( d_{ij} = d_\phi(\tau_i, \tau_j) \) is the estimated dissimilarity between trajectories of agent \( i \) and \( j \). Such formulation makes sure that dissimilarity \( d \) is high only when mutual information \( I \) is low, so that the set of learned roles is compact but diverse, which help solve the given task efficiently. Formally, the following learning objective encourages sub-task specialization:

\[
\begin{align*}
\min_{\theta, \phi, \xi} & \quad D_\phi \quad 2.0 \\
\text{subject to} & \quad I(\rho^t_i; \tau_j) + d_\phi(\tau_i, \tau_j) > U, \forall i \neq j,
\end{align*}
\]

where \( U \) controls the compactness of the role representation.

In practice, we separately carry out min-max normalization on \( I \) and \( d \) to scale their values to \([0, 1]\) and set U to 1. Relaxing the matrix norm \( \cdot \quad 2.0 \) with the Frobenius norm, we can get the optimization objective for minimizing:

\[
D_\phi \quad F = \min_{i \neq j} \left( I(\rho^t_i; \tau_j) + d_\phi(\tau_i, \tau_j) \right),
\]

However, as estimating and optimizing the mutual information term are intractable, we use the variational posterior estimator introduced in Sec. 3.1 to construct an upper bound, serving as the second regularizer of ROMA:

\[
\mathcal{L}(\rho^t, \phi, \xi) = \mathbb{E}(\tau_t \sim \rho^t, \alpha^t, \alpha^t \sim \rho^t) \quad D_\phi \quad F
\]

3.3. Overall Optimization Objective

We have introduced optimization objectives for learning roles to be identifiable and and specialized. Apart from these regularizers, all the parameters in the framework are updated by gradients induced by the standard TD loss of reinforcement learning. As shown in Fig. 2, to compute the global TD loss, individual utilities are fed into a mixing network whose output is the estimation of global action-value \( Q_{\text{tot}} \). In this paper, our ROMA implementation uses the mixing network introduced by QMIX (Rashid et al., 2018) (see Appendix D) for its monotonic approximation, but it can be easily replaced by other mixing methods. The parameters of the mixing network are conditioned on the global state \( s \) and are generated by a hyper-net parameterized by \( \theta_m \). Therefore, the final learning objective of ROMA is:

\[
\mathcal{L}(\theta) = \mathcal{L}_{\text{TD}}(\theta) + \lambda_{I} \mathcal{L}_{\text{P}}(\rho, \xi) + \lambda_{D} \mathcal{L}_{\text{D}}(\theta, \xi, \phi),
\]

where \( \theta = (\theta_p, \xi, \phi, \theta_h, \theta_m) \), \( \lambda_I \) and \( \lambda_D \) are scaling factors, and \( \mathcal{L}_{\text{TD}}(\theta) = [\tau + \gamma \max_{\alpha^t'} Q_{\text{tot}}(s', \alpha^t; \theta') - Q_{\text{tot}}(s, \alpha; \theta)]^2 \) (\( \theta^* \) are the parameters of a periodically updated target network). In our centralized training with decentralized execution framework, only the role encoder, the role decoder, and the individual utility networks are used when execution.
ROMA: Role-Oriented Multi-Agent Reinforcement Learning

4. Related Works

The emergence of role has been documented in many natural systems, such as bees (Jeanson et al., 2005), ants (Gordon, 1996), and humans (Butler, 2012). In these systems, the role is closely related to the division of labor and is crucial to the improvement of labor efficiency. Many multi-agent systems are inspired by these natural systems. They decompose the task, make agents with the same role specialize in certain sub-tasks, and thus reduce the design complexity (Wooldridge et al., 2000; Omicini, 2000; Padgham & Winikoff, 2002; Pavón & Gómez-Sanz, 2003; Cossentino et al., 2005; Zhu & Zhou, 2008; Spanoudakis & Moraitis, 2010; DeLoach & Garcia-Ojeda, 2010; Bonjean et al., 2014). These methodologies are designed for tasks with a clear structure, such as software engineering (Bresciani et al., 2004). Therefore, they tend to use predefined roles and associated responsibilities (Lhaksmana et al., 2018). In contrast, we focus on how to implicitly introduce the concept of roles into general multi-agent sequential decision making under dynamic and uncertain environments.

Deep multi-agent reinforcement learning has witnessed vigorous progress in recent years. COMA (Foerster et al., 2018), MADDPG (Lowe et al., 2017), PR2 (Wen et al., 2019), and MAAC (Iqbal & Sha, 2019) explore multi-agent policy gradients. Another line of research focuses on value-based multi-agent RL, and value-function factorization is the most popular method. VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and QTRAN (Son et al., 2019) have progressively enlarged the family of functions that can be represented by the mixing network. NDQ (Wang et al., 2020b; Kang et al., 2020), the emergence of fairness (Jiang & Lu, 2019), and the emergence of tool usage (Baker et al., 2020) provide a deep learning perspective in understanding both natural and artificial multi-agent systems.

To learn diverse and identifiable roles, we propose to optimize the mutual information between individual roles and trajectories. A recent work studying multi-agent exploration, MAVEN (Mahajan et al., 2019), uses a similar objective. Different from ROMA, MAVEN aims at committed exploration. This difference in high-level purpose leads to many technical distinctions. First, MAVEN optimizes the mutual information between the joint trajectory and a latent variable conditioned on a Gaussian or uniform random variable to encourage diverse joint trajectory. Second, apart from the mutual information objective, we propose a novel regularizer to learn specialized roles, while MAVEN adopts a hierarchical structure and encourages the latent variable to help get more environmental rewards. We empirically compare ROMA with MAVEN in Sec. 5. More related works will be discussed in Appendix D.

5. Experiments

Our experiments aim to answer the following questions: (1) Whether the learned roles can automatically adapt in dynamic environments? (Sec. 5.1.) (2) Can our method promote sub-task specialization? That is, agents with similar responsibilities have similar role embedding representations, while agents with different responsibilities have role embedding representations far from each other. (Sec. 5.1, 5.3.) (3) Can such sub-task specialization improve the performance of multi-agent reinforcement learning algorithms? (Sec. 5.2.) (4) How do roles evolve during training, and how do they influence team performance? (Sec. 5.4.) (5) Can the dissimilarity model \( d_\phi \) learn to measure the dissimilarity...
between agents’ trajectories? (Sec. 5.4.) Videos2 of our experiments and the code3 are available online.

**Baselines** We compare our methods with various baselines shown in Table 1. In particular, we carry out the following ablation studies: (i) We separately omit each (or both) of the two role-learning objectives ($L_I$ and $L_D$) while leaving the other parts of ROMA unchanged. These three ablations are designed to highlight the contribution of each of the proposed regularizers. (ii) QMIX-NPS. The same as QMIX (Rashid et al., 2018), but agents do not share parameters. Our method achieves adaptive learning sharing, and comparison against QMIX (parameters are shared among agents) and QMIX-NPS tests whether this flexibility can improve learning efficiency. (iii) QMIX-LAR, QMIX with a similar number of parameters with our framework, which can test whether the superiority of our method comes from the increase in the number of parameters.

We carry out a grid search over the loss coefficients $\lambda_I$ and $\lambda_D$, and fix them at $10^{-4}$ and $10^{-2}$, respectively, across all the experiments. The dimensionality of latent role space is set to 3, so we did not use any dimensionality reduction techniques when visualizing the role embedding representations. Other hyperparameters are also fixed in our experiments, which are listed in Appendix B.1. For ROMA, We use elementary network structures (fully-connected networks or GRU) for the role encoder, role decoder, and trajectory encoder. The details of the architecture of our method and baselines can be found in Appendix B.

| Table 1. Baseline algorithms. |
|-------------------------------|-----------|
| **Alg.** | **Description** |
| IQL | Independent Q-learning |
| COMA | Foerster et al. (2018) |
| QMIX | Rashid et al. (2018) |
| QTRAN | Son et al. (2019) |
| MAVEN | Mahajan et al. (2019) |
| $L_I$ | ROMA without $L_I$ and $L_D$ |
| $L_I + L_D$ | ROMA without $L_I$ |
| Ablations | QMIX-NPS |
| | QMIX with similar number of parameters with ROMA |

### 5.1. Dynamic Roles

Answering the first and second questions, we show snapshots in an episode played by ROMA agents on the StarCraft II micromanagement benchmark (SMAC) map 10m vs. 11m, where 10 Marines face 11 enemy Marines. As shown in Fig. 3 (the role representations at $t=27$ are presented in Fig. 1), although observations contain much information, such as positions, health points, shield points, states of ally and enemy units, etc., the role encoder learns to focus on different parts of the observations according to the dynamically changed situations. At the beginning ($t=1$), agents need to form a concave arc to maximize the number of agents whose shoot range covers the front line of enemies. ROMA learns to allocate roles according to agents’ relative positions so that agents can quickly form the offensive formation using
specialized policies. In the middle of the battle, one important tactic is to protect the injured ranged units. Our method learns this maneuver and roles cluster according to the remaining health points ($t=8, 19, 27$). Healthiest agents have role representations far from those of other agents. Such representations result in differentiated strategies: healthiest agents move forward to take on more firepower while other agents move backward, firing from a distance. In the meantime, some roles also cluster according to positions (agents 3 and 8 when $t=19$). The corresponding behaviors are agents with different roles fire alternatively to share the firepower. We can also observe that the role representations of dead agents aggregate together, representing a special group with an increasing number of agents during the battle.

These results demonstrate that our method learns dynamic roles and roles cluster clearly corresponding to automatically detected sub-tasks, in line with implicit constraints of the proposed optimization objectives.

5.2. Performance on StarCraft II

To test whether these roles and the corresponding sub-task specialization can improve learning efficiency, we test our method on the StarCraft II micromanagement (SMAC) benchmark (Samvelyan et al., 2019). This benchmark consists of various maps which have been classified as easy, hard, and super hard. We compare ROMA with algorithms shown in Table 1 and present results for one easy map (2s3z), three hard maps (5m_vs_6m, 8m_vs_9m & 10m_vs_11m), and two super hard maps (MMM2 & 27m_vs_30m). Although SMAC benchmark is challenging, it is not specially designed to test performance in tasks with many agents. We thus introduce three new SMAC maps to test the scalability of our method, which are described in detail in Appendix C.

For evaluation, all experiments in this section are carried out with 5 different random seeds, and results are shown with a 95% confidence interval. Among these maps, four maps, MMM2, 6s4z_vs_10b30z, 6z4b, and 10z5b_vs_2z3s, feature heterogeneous agents, and the others have homogeneous agents. Fig. 4 shows that our method yields substantially better results than all the alternative approaches on both homogeneous and heterogeneous maps (additional plots can be found in Appendix C.1). MAVEN overcomes the negative effects of QMIX’s monotonicity constraint on exploration. However, it performs less satisfactorily than QMIX on most maps. We believe this is because agents start engaging in the battle immediately after spawning in SMAC maps, and exploration is not the critical factor affecting performance.

Ablations We carry out ablation studies, comparing with the ablations shown in Table 1 and present results on three maps: MMM2 (heterogeneous), 10z5b_vs_2s3z, and 10m_vs_11m (homogeneous) in Fig. 5 and 6. The superiority of our method against $\mathcal{L}_{TD}$ highlights the contribution of the proposed regularizers – $\mathcal{L}_{TD}$ performs even worse than QMIX on two of the three maps. By comparing ROMA with $\mathcal{L}_{TD} + \mathcal{L}_I$ and $\mathcal{L}_{TD} + \mathcal{L}_D$, we can conclude that the specialization loss $\mathcal{L}_D$ is more important in terms of performance improvements. Introducing $\mathcal{L}_I$ can make training more stable (for example, on the map 10m_vs_11m), but optimizing $\mathcal{L}_I$ alone can only slightly improve the performance. These observations support the claim that sub-task specialization can improve labor efficiency.
Comparison between QMIX-NPS and QMIX demonstrates that parameter sharing can, as documented (Foerster et al., 2018; Rashid et al., 2018), speed up training. As discussed in the introduction, both these two paradigms may not get the best possible performance. In contrast, our method provides a dynamic learning sharing mechanism – agents committed to a certain responsibility have similar policies. The comparison of the performance of ROMA, QMIX, and QMIX-NPS proves that such sub-task specialization can indeed improve team performance. What’s more, comparison of ROMA against QMIX-LAR proves that the superiority of our method does not depend on the larger number of parameters.

The performance gap between ROMA and ablations is more significant on maps with more than ten agents. This observation supports discussions in previous sections – the emergence of role is more likely to improve the labor efficiency in larger populations.

5.3. Role Embedding Representations

To explain the superiority of ROMA, we present the learned role embedding representations for three maps in Fig. 7. Roles are representative of automatically discovered sub-tasks in the learned winning strategy. In the map of 6a4z_v8_10b30z, ROMA learns to sacrifice Zealots 9 and 7 to kill all the enemy Banelings. Specifically, Zealots 9 and 7 will move to the frontier one by one to minimize the splash damage, while other agents will stay away and wait until all Banelings explode. Fig. 7(a) shows the role embedding representations while performing the first sub-task where agent 9 is sacrificed. We can see that the role of Zealot 9 is quite different from those of other agents. Correspondingly, the strategy at this time is agent 9 moving rightward while other agents keep still. Detailed analysis for the other two maps can be found in Appendix C.2.

5.4. Emergence and Evolution of Roles

We have shown the learned role representations and performance of our method, but the relationship between roles and performance remains unclear. To make up for this shortcoming, we visualize the emergence and evolution of roles during the training process on the map MMM2 (heterogeneous) and 10m_vs_11m (homogeneous). We discuss the results on MMM2 here and defer analysis of 10m_vs_11m to Appendix C.3.

In MMM2, 1 Medivac, 2 Marauders, and 7 Marines are faced with a stronger enemy team consisting of 1 Medivac, 3 Marauders, and 8 Marines. Among the three involved unit types, Medivac is the most special one for that it can heal the injured units. In Fig. 8, we show one of the learning curves of ROMA (red) and the role representations at the first environment step at three different stages. When the training begins (T=0), roles are random, and the agents are exploring the environment to learn the basic dynamics and the structure of the task. By T=6M, ROMA has learned that the responsibilities of the Medivac are different from those of Marines and Marauders. The role, and correspondingly, the policy of the Medivac becomes quite different (Fig. 8 middle). Such differentiation in behaviors enables agents to start winning the game. Gradually, ROMA learns that...
Marines and Marauders have dissimilar characteristics and should take different sub-tasks, indicated by the differentiation of their role representations (Fig. 8 right). This further specialization facilitates the performance increase between 6M and 10M. After \( T = 10M \), the responsibilities of roles are clear, and, as a result, the win rate gradually converges (Fig. 4 top left). For comparison, ROMA without \( L_I \) and \( L_D \) can not even win once on this challenging task (\( L_{TD} \) in Fig. 6-left). These results demonstrate that the gradually specialized roles are indispensable in team performance improvement.

Moreover, we find that the learned dissimilarity model \( d_\phi \) introduced in Sec. 3.2 provides an empirical evaluation for identifying new roles. We use the map \#9992 as an example, where, as we discussed above, the learned roles of agents are characterized by their unit types. After scaling to \([0, 1]\), the learned dissimilarity between trajectories of agents with different unit types is close to 0.96, while the learned dissimilarity between trajectories of agents with the same unit type is around 0.08. These results indicate that an appropriate threshold can be used to decide when an individual behavior (trajectory) can be assigned the terminology role.

In summary, our experiments demonstrate that ROMA can learn dynamic, identifiable, versatile, and specialized roles that effectively decompose the task. Drawing support from these emergent roles, our method significantly pushes forward the state of the art of multi-agent reinforcement learning algorithms.

6. Closing Remarks

We have introduced the concept of roles into deep multi-agent reinforcement learning by capturing the emergent roles and encouraging them to specialize on a set of automatically detected sub-tasks. Such deep role-oriented multi-agent learning framework provides another perspective to explain and promote cooperation within agent teams, and implicitly draws connection to the division of labor, which has been practiced in many natural systems for long.

To our best knowledge, this paper is making a first attempt at learning roles via deep reinforcement learning. The gargantuan task of understanding the emergence of roles, the division of labor, and interactions between more complex roles in hierarchical organization still lies ahead. We believe that these topics are basic and indispensable in building effective, flexible, and general-purpose multi-agent systems and this paper can help tackle these challenges.

Acknowledgements

We gratefully acknowledge Jin Zhang for his valuable discussions. We also would like to thank the reviewers for their detailed and constructive feedback. This work is partially supported by the sponsorship of Guoqiang Institute at Tsinghua University.

References


Cossentino, M., Gaglio, S., Sabatucci, L., and Seidita, V. The passi and agile passi mas meta-models compared with a unifying proposal. In International Central and


Smith, A. The wealth of nations [1776], 1937.


