ROMA: Multi-Agent Reinforcement Learning with Emergent Roles

TonghanWang*, Heng Dong*, Victor Lesser, Chongjie Zhang
37th ICML 2020

2024. 07. 22

AI Robotics KR, 에이전트브레인스토밍
Minkyoung Kim

Abstract

- 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.
- Existing role-based methods use prior domain knowledge.
- Existing role-based methods predefine role structures and behaviors.

- Role-oriented MARL framework(ROMA)
  - The first attempt at learning roles via deep reinforcement learning
  - Agents with similar roles tend to share their learning and to be specialized on certain sub-tasks.
  - **Stochastic role embedding space**
    - by introducing two novel regularizers and conditioning individual policies on role
  - Can learn **specialized, dynamic, and identifiable roles**

Introduction

- MARL methods adapt a simple mechanism
  that all agents share and learn a decentralized value or policy network

  Limit: such simple sharing is often not effective for many complex multi-agent tasks

  →It is heavy burden for a single shared policy to represent and learn all required skills

  Limit: 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

- How can we give full play to agent's specialization and dynamic sharing for improving learning efficiency?

Introduction

- Natural concept that come to mind is the role
  - Comprehensive pattern of behavior, often specialized in some tasks

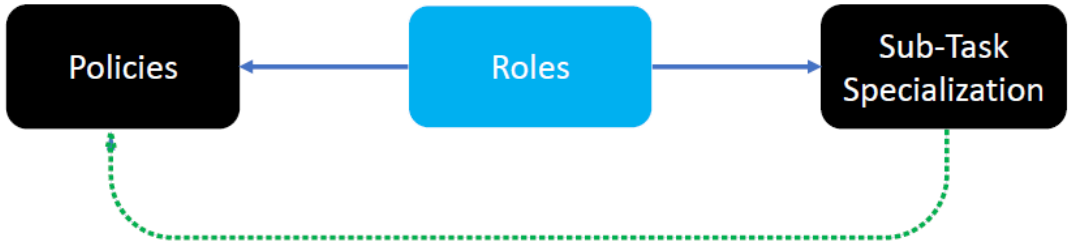
- Agents with similar role will show similar behavior, and thus can share their experiences to improve performance

- Researchers have also introduced the concept of role into MAS(1999~2018)
  - The complexity of agent design is reduced via task decomposition
    by defining roles associated with responsibilities made up of a set of sub-tasks
  - Limit : Exploit prior domain knowledge to decompose tasks and predefine the responsibilities of each role

  → Prevents role-based MAS from being dynamic and adaptive to uncertain environments

Introduction

- **ROMA(Role-oriented Multi-agent reinforcement learning)**
  - Enable agents with similar responsibilities to share their learning.
  - by ensuring that agents with **similar roles** have **both similar policies and responsibilities.**
  - **Roles and decentralized policy**
    - Generating role embedding(stochastic latent variables) by a role encoder conditioned on local observations
    - Conditioning agent's policies on individual roles
  - **Role and responsibilities**
    - Two regularizers to enable roles to be:
      - identifiable by behaviors
      - specialized in certain sub-tasks



  - Promotes the emergence and specialization of roles
  - Provides and adaptive learning sharing mechanisms for efficient multi-agent policy sharing
  - Experiments ROMA on SMAC

Background

- Fully cooperative multi-agent task, Dec-POMDP
- $G = < I, S, A, P, R, \Omega, O, n, \gamma >$
  - A : finite action set
  - I : finite set of *n* agents
  - $\gamma \in [0,1]$, discount factor
  - $s \in S$, true state of the environment
  - Agent *i* only has access to an obversions $o_i \in \Omega$,
    - drawn according to the observation function $O(s, i)$
  - Each agent *i* has history $\tau_i \in T \equiv (\Omega \times A)^*$
  - Each agent *i* select an action $a_i \in A$, forming a joint action $\boldsymbol{a} \in A^n$
  - Transition function $P(s'|s, \boldsymbol{a})$
  - Shared reward $\mathrm{r} = R(s, \boldsymbol{a})$
  - Joint policy π induces a joint action-value function

$$Q_{tot}^{\boldsymbol{\pi}}(s,\boldsymbol{a}) = \mathbb{E}_{s_{0:\infty}, \boldsymbol{a}_{0:\infty}} \left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| \, s_0 = s, \boldsymbol{a}_0 = \boldsymbol{a}, \boldsymbol{\pi} \right].$$

- Existing CTDE methods learn a shared local value or policy network for agent.

  → Not sufficient for learning complex task,
  where diverse responsibilities or skills are required to achieve goals

Method

- ROMA adopts the CTDE paradigm
  - During Learning
    - Learn local Q-value functions for agents.
    - Local Q-value functions are fed into a mixing network to compute a global TD loss for centralized training.
  - During Execution
    - The mixing network will be removed
    - Each agent will act based on its local policy
    - Agent's value functions or policies are dependent on their roles

[QMIX]

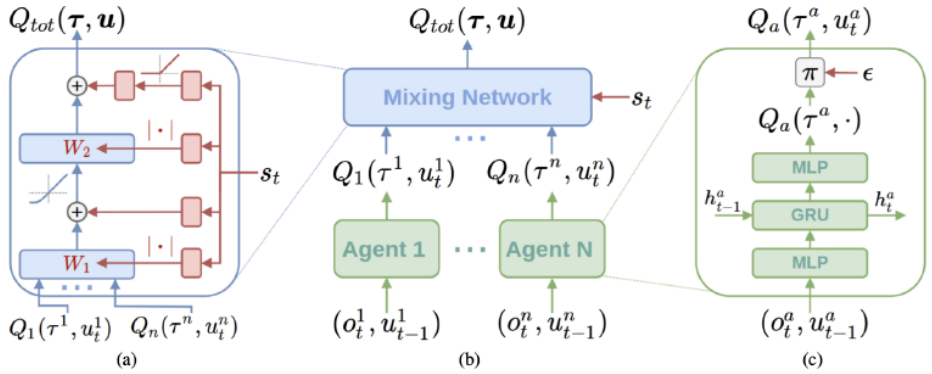
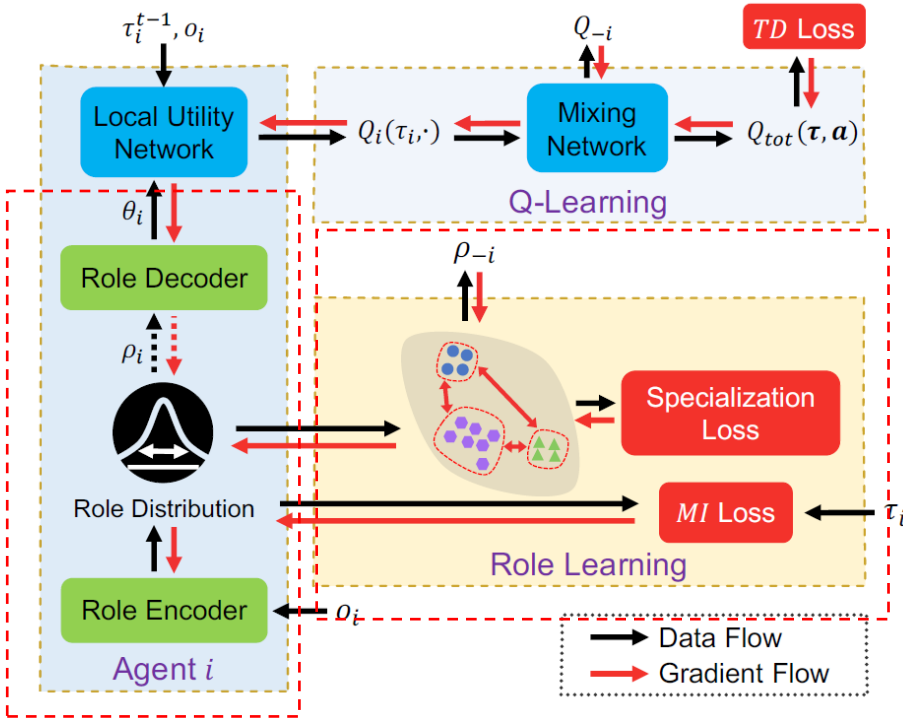


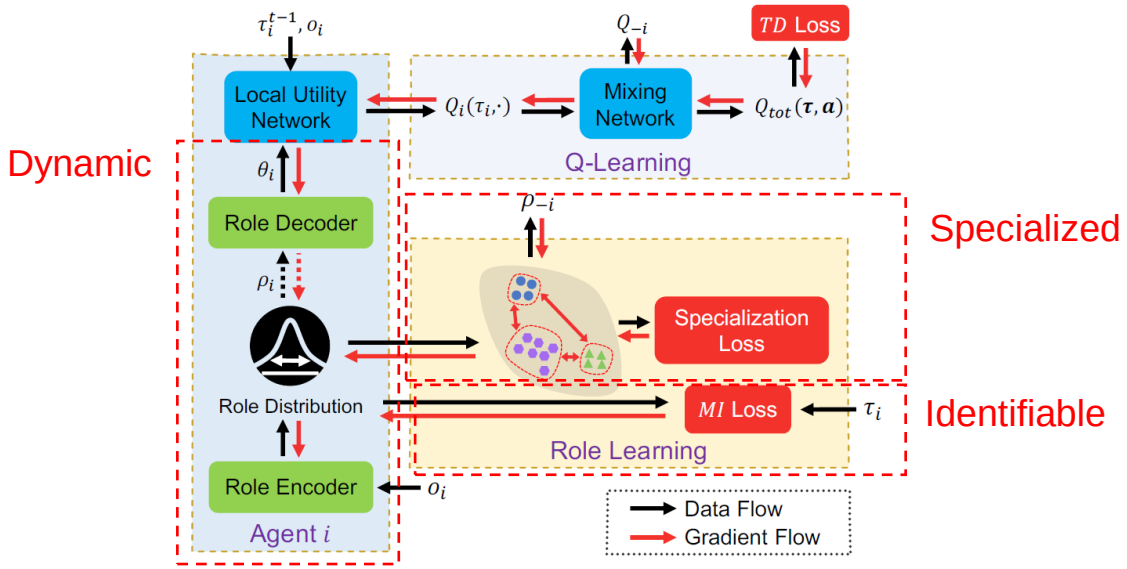
Figure 13. The framework of QMIX, reproduced from the original paper (Rashid et al., 2018). (a) The architecture of the mixing network (blue), whose weights and biases are generated by a hyper-net (red) conditioned on the global state. (b) The overall QMIX structure. (c) Local utility network structure.



[ROMA]

Method

- To enable efficient and effective shared learning among agents with similar behaviors,
- ROMA learn roles that are:
  - **Dynamic**
    - An agent's role can automatically adapt to the dynamics of the environment
  - **Identifiable**
    - The role of an agent contains enough information about its behaviors
  - **Specialized**
    - Agents with similar roles are expected to specialize in similar sub-tasks

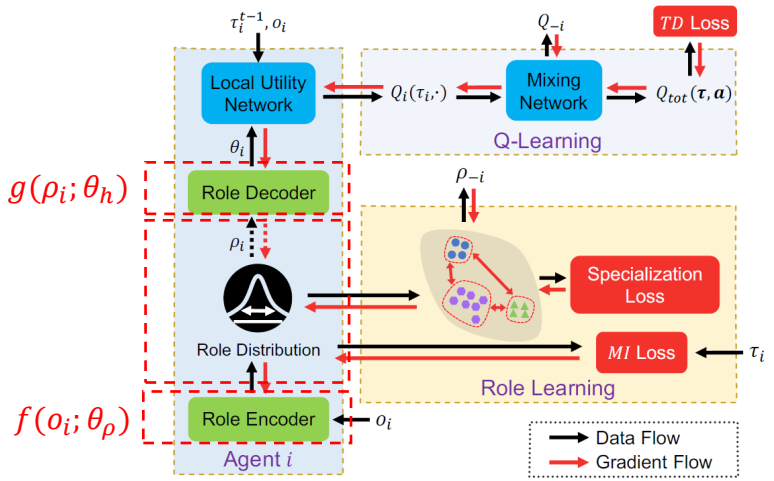


Method

- **Dynamic**
  : Conditioning roles on local observations enable roles to be responsible to changes in the environments
  - Each agent *i* has local utility function, whose parameters θ_i are conditioned on its role ρ_i
  - ROMA conditions an agent's role on its local observations
  - **Role distribution**
    - To learn roles, encode roles in a stochastic embedding space ρ_i, drawn from a multivariate Gaussian distribution $N(\mu_{\rho_i}, \sigma_{\rho_i})$.
  - **Role Encoder**
    - ROMA uses a trainable neural network f to learn the parameters of Gaussian distribution of the role
    - θ_ρ *are parameters of* f

$$(\boldsymbol{\mu}_{\rho_i}, \boldsymbol{\sigma}_{\rho_i}) = f(o_i; \theta_\rho),$$
$$\rho_i \sim \mathcal{N}(\boldsymbol{\mu}_{\rho_i}, \boldsymbol{\sigma}_{\rho_i}),$$

  - **Role Decoder**
    - generate the parameters for the individual policy, θ_i
    - a hyper-network $g(\rho_i; \theta_h)$
      - Sampled role ρ_i is then fed into hyper-network



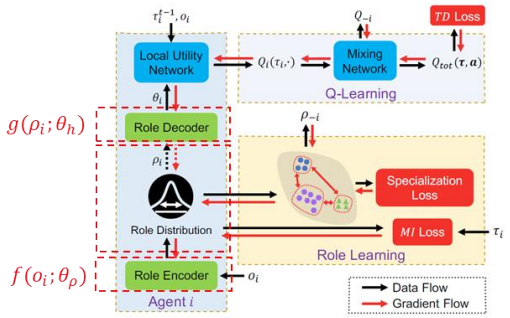
Method

- Dynamic
  - Introducing latent role embedding and conditioning individual policies on this embedding does not automatically generate roles with desired properties
  - may cause roles to change quickly, making learning unstable

- **Identifiable Roles**
  : Minimizing L_I enables role to contain enough information about **long-term** behaviors
  - propose to learn roles that are identifiable by agent's long-term behavior
  - by maximizing $I(\tau_i; \rho_i \mid o_i)$, conditional mutual information between the individual trajectory and role given the current observation.
    - introduce a variational posterior estimator to derive a tractable lower bound for the mutual information objective (proof. Appendix A.1)
    - q_ξ is the variational estimator with ξ
      - **trajectory encoder**(uses GRU)
      - encode an agent's history of observations and actions
  - loss function L_I

$$I(\rho_i^t; \tau_i^{t-1} | o_i^t) \geq \mathbb{E}_{\rho_i^t, \tau_i^{t-1}, o_i^t} \left[\log \frac{q_\xi(\rho_i^t | \tau_i^{t-1}, o_i^t)}{p(\rho_i^t | o_i^t)} \right]$$

$$\mathcal{L}_I(\theta_\rho, \xi) = \mathbb{E}_{(\tau_i^{t-1}, o_i^t) \sim D} \left[D_{\mathrm{KL}} [p(\rho_i^t | o_i^t) \| q_\xi(\rho_i^t | \tau_i^{t-1}, o_i^t)] \right]$$



Method

- Identifiable Roles
  - does not explicitly ensure agents with similar behaviors to have similar role embedding

- **Specialized Roles(Cont.)**
  : sub-task specialization, which is the critical component to share learning and improve efficiency
  - for any two agent, we expect that:
    - **Either** they have similar roles
    - **or** they have quite different behaviors
  - to encourage two agent *i* and *j* to have similar role, we can maximize $I(\tau_i; \rho_j | o_j)$
    - mutual information between the role of agent *i* and the trajectory of agent *j*.

$$I(\rho_i^t; \tau_j^{t-1} | o_j^t)$$

Method

- **Specialized Roles**
  - learnable dissimilarity model $d_\phi: \mathrm{T} \times \mathrm{T} \rightarrow \mathbb{R}$
    - estimated dissimilarity between trajectories of agent *i* and *j*
    - trainable neural network taking two trajectories as input
    - seek to maximize $I(\tau_i; \rho_j | o_j) + d_\phi(\tau_i, \tau_j)$
      - while the number of non-zero elements in the matrix $D_\phi = (d_{ij}),\ d_{ij} = d_\phi(\tau_i, \tau_j)$
      - dissimilarity *d* is high only when mutual information *I* is low
  - loss function L_D
    - Detailed derivation Appendix A.2

$$\begin{aligned} &\underset{\theta_\rho, \xi, \phi}{\text{minimize}} \quad \|D_\phi^t\|_{2,0} \qquad (4) \\ &\text{subject to} \quad I(\rho_i^t; \tau_j^{t-1} | o_j^t) + d_\phi(\tau_i^{t-1}, \tau_j^{t-1}) > U, \forall i \neq j, \end{aligned}$$

$$\mathcal{L}_D(\theta_\rho, \phi, \xi) = \mathbb{E}_{(\boldsymbol{\tau}^{t-1}, \boldsymbol{o}^t) \sim \mathcal{D}, \boldsymbol{\rho}^t \sim p(\boldsymbol{\rho}^t | \boldsymbol{o}^t)} \left[\|D_\phi^t\|_F - \sum_{i \neq j} \min\{q_\xi(\rho_i^t | \tau_j^{t-1}, o_j^t) + d_\phi(\tau_i^{t-1}, \tau_j^{t-1}), U\} \right]$$

Method

- Overall Optimization Objectives
  - all parameters in farmwork are updated by gradient induced by the standard TD loss

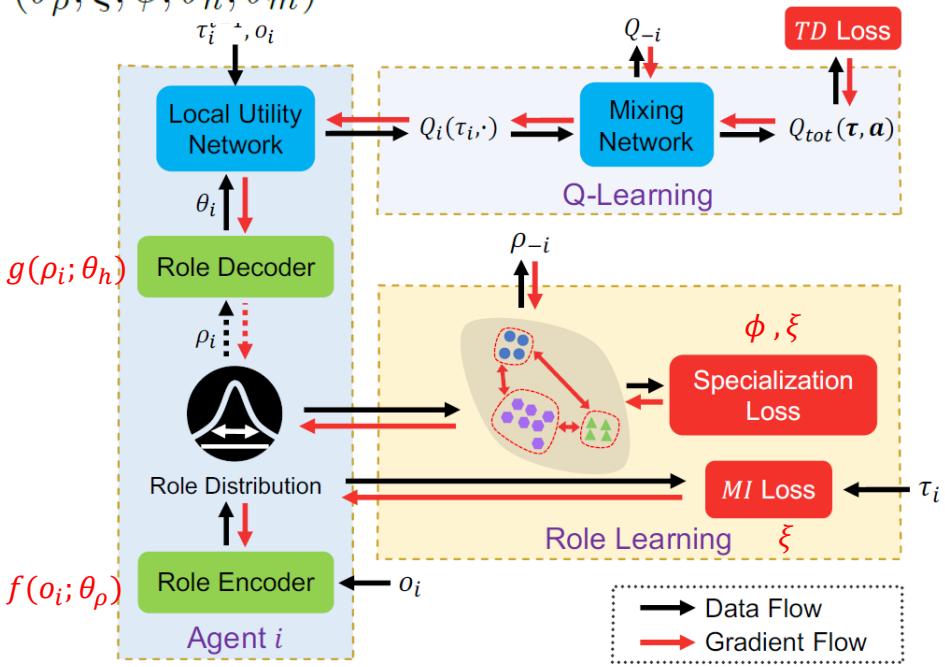
$$\mathcal{L}(\theta) = \mathcal{L}_{TD}(\theta) + \lambda_I \mathcal{L}_I(\theta_\rho, \xi) + \lambda_D \mathcal{L}_D(\theta_\rho, \xi, \phi),$$

standard loss
θ_h: role decoder
θ_m: mixing network

identifiable loss
θ_ρ: role encoder
ξ : trajectory encoder

specialization loss
θ_ρ: role encoder
ξ : trajectory encoder
ϕ: dissimilarity model

$$\theta = (\theta_\rho, \xi, \phi, \theta_h, \theta_m)$$



specialization loss
λ_I: 10^{-4}
λ_D: 10^{-2}

Experiments

- Whether the learned roles can automatically adapt in dynamic environment?
- Can our method promote sub-task specialization?
  - agents with similar responsibility have similar role embedding representation.
- Can such sub-task specialization improve the performance of multi-agent reinforcement learning algorithms?
- How do roles evolve during training, and how do they influence team performance?
- Can the dissimilarity model d_ϕ learn to measure the dissimilarity between agent's trajectories?

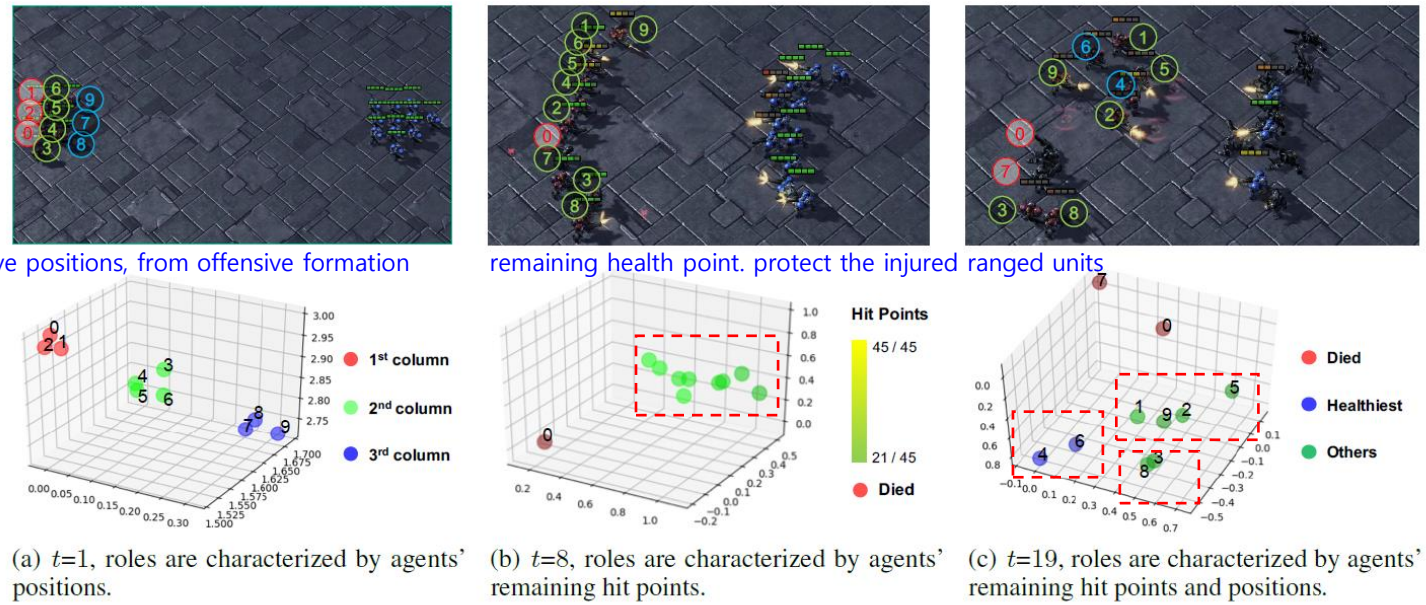
Experiments

Table 1. Baseline algorithms.

	Alg.	Description
Related Works	IQL	Independent Q-learning
	COMA	Foerster et al. (2018)
	QMIX	Rashid et al. (2018)
	QTRAN	Son et al. (2019)
	MAVEN	Mahajan et al. (2019)
Ablations	$\mathcal{L}_{TD}$	ROMA without $\mathcal{L}_I$ and $\mathcal{L}_D$
	$\mathcal{L}_{TD} + \mathcal{L}_I$	ROMA without $\mathcal{L}_D$
	$\mathcal{L}_{TD} + \mathcal{L}_D$	ROMA without $\mathcal{L}_I$
	QMIX-NPS	QMIX without parameter sharing among agents
	QMIX-LAR	QMIX with similar number of parameters with ROMA

Experiments

- Whether the learned roles can automatically adapt in dynamic environment?
- Can our method promote sub-task specialization?
  - agents with similar responsibility have similar role embedding representation.



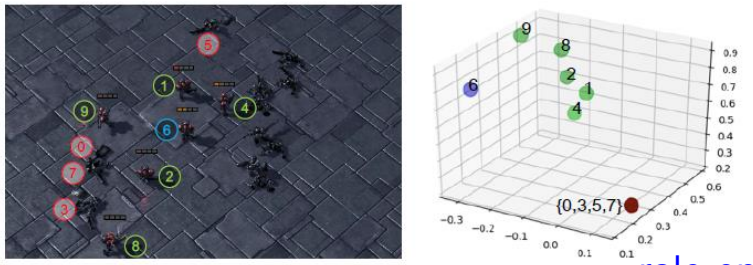
(a) t=1, roles are characterized by agents' positions.

(b) t=8, roles are characterized by agents' remaining hit points.

(c) t=19, roles are characterized by agents' remaining hit points and positions.

Figure 3. Dynamic role adaptation during an episode (means of the role distributions, μ_{ρ_i}, are shown, without using any dimensionality reduction techniques). The *role encoder* learns to focus on different parts of observations according to the automatically discovered demands of the task. The role-induced strategy helps (a) quickly form the offensive arc when t=1; (b) protect injured agents when t=8; (c) protect dying agents and alternate fire when $t = 19$.

t = 27



role encoder

Figure 1. Visualization of our learned role representations at a timestep. The blue agent has the maximum health, while the red ones are dead. The corresponding policy is that `agent 6` moves towards enemies to take on more firepower, so that more seriously injured agents are protected. Roles can change adaptively and will aggregate according to responsibilities that are compatible with individual characteristics, such as location, agent type, health, etc.

healthiest agents: move forward to take on more firepower
others: move backward

Experiments

- Can such sub-task specialization improve the performance of multi-agent reinforcement learning algorithms?

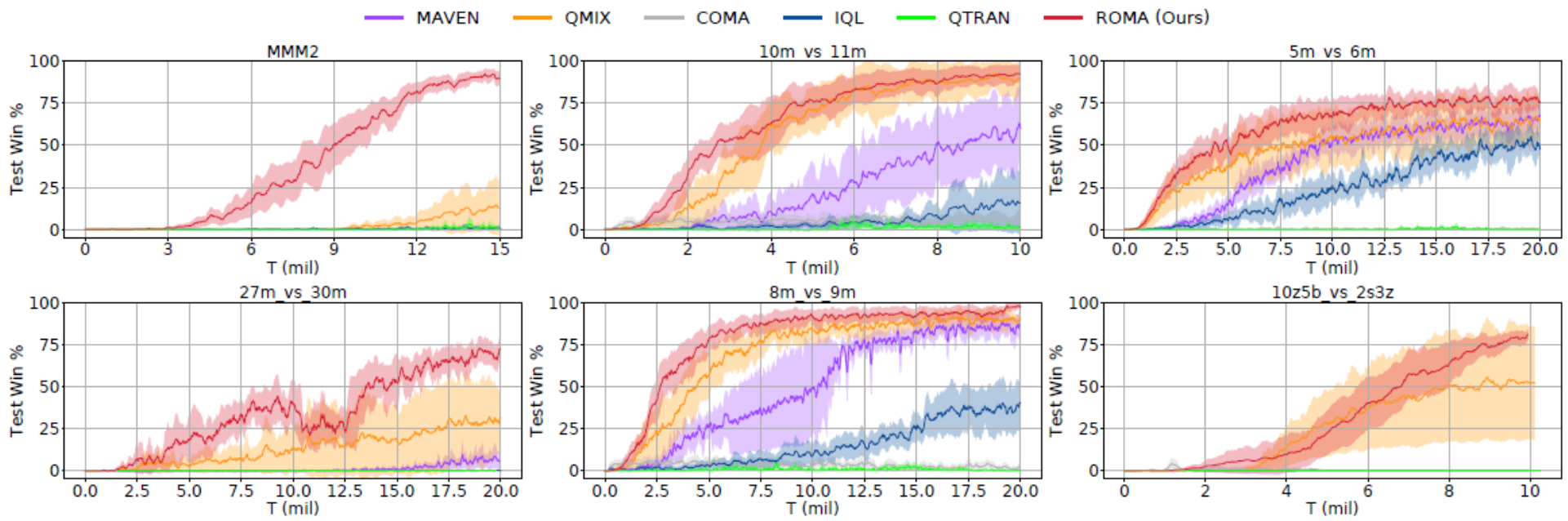


Figure 4. Comparison of our method against baseline algorithms. Results for more maps can be found in Appendix C.1.

Experiments

- Can such sub-task specialization improve the performance of multi-agent reinforcement learning algorithms?

L_{TD}: superiority of proposed regularizer
L_D is more important in terms of performance improvements

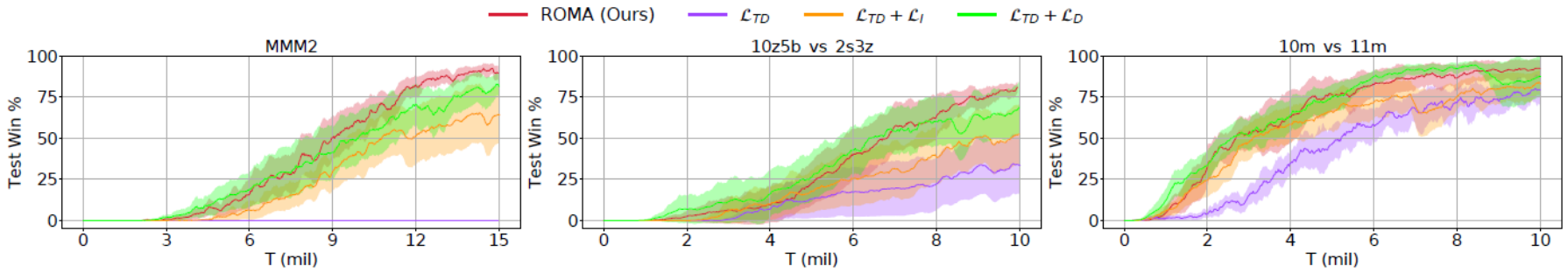


Figure 5. Ablation studies regarding the two role-learning losses.

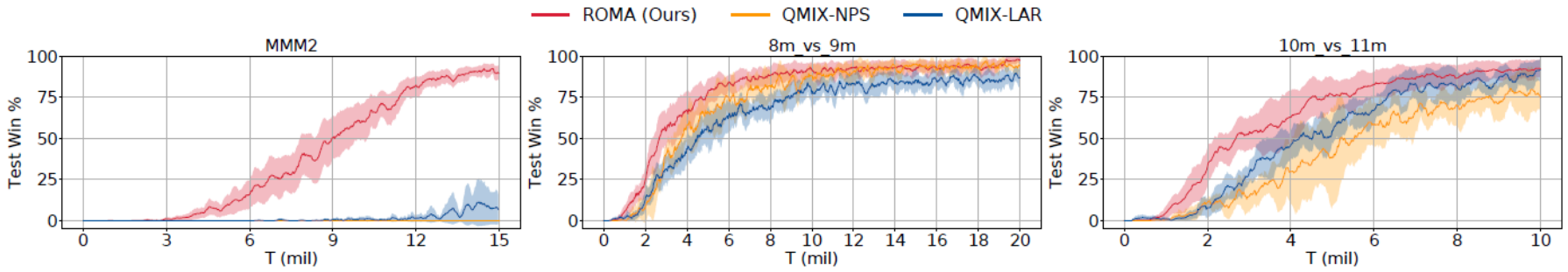
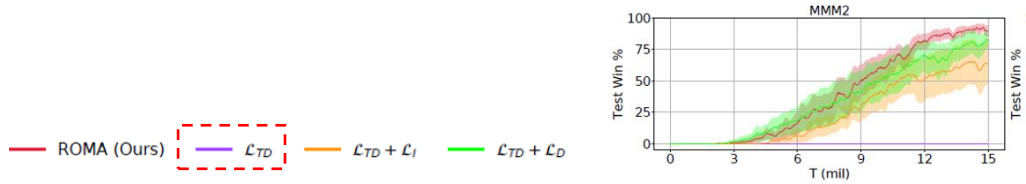


Figure 6. Comparison of our method against ablations.

QMIX-LAR: superiority of ROMA does not depend on the larger number of parameter

Experiments



- How do roles evolve during training, and how do they influence team performance?

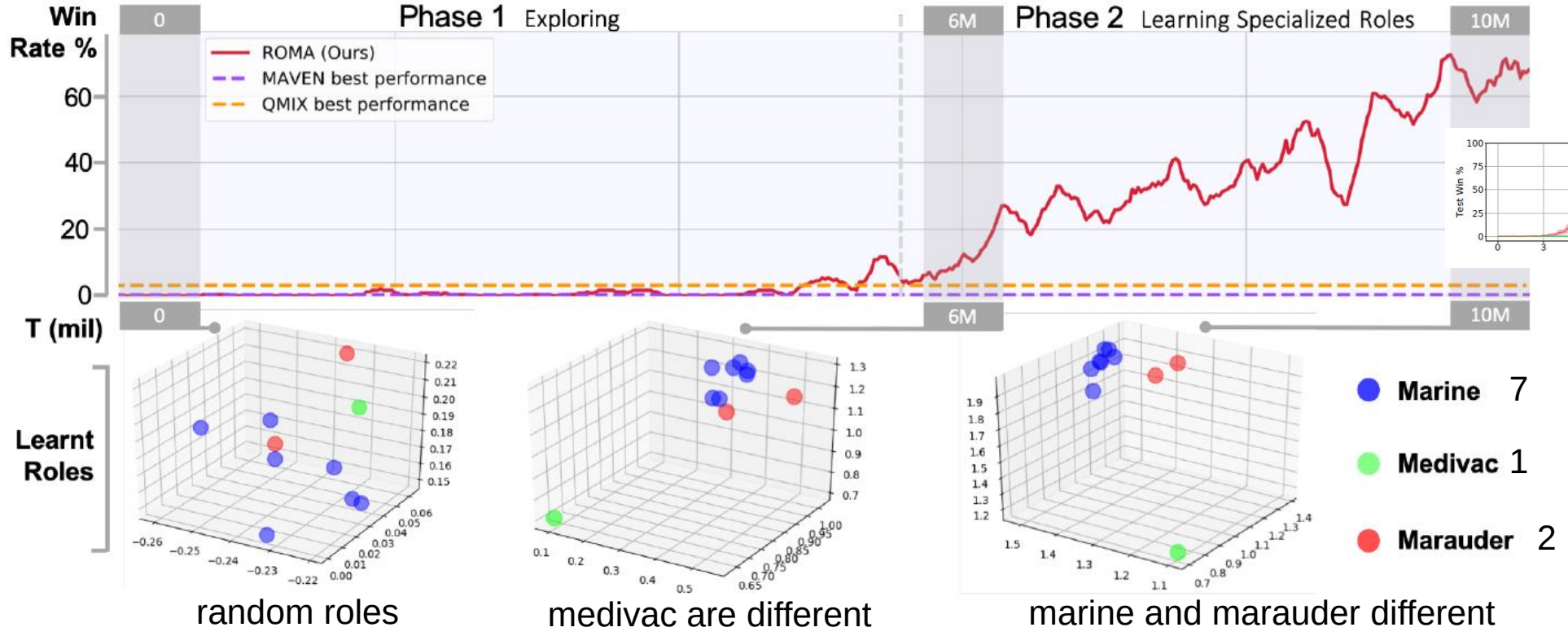


Figure 8. Role emergence and evolution on the map MMM2 (role representations at time step 1 are shown) during training (means of the role distributions, μ_{ρ_i}, are shown, without using any dimensionality reduction techniques). The emergence and specialization of roles is closely connected to the improvement of team performance. Agents in MMM2 are heterogeneous, and we show role evolution process in a homogeneous team in Appendix C.3.

Experiments

- Can the dissimilarity model d_ϕ learn to measure the dissimilarity between agent's trajectories?

Table 2. The mean and standard deviation of the learned dissimilarities d_ϕ between agents' trajectories on the map `MMM2`.

Between different unit types	0.9556 ± 0.0009
Between the same unit type	0.0780 ± 0.0019

Reference

- https://slideslive.com/38927654/roma-multiagent-reinforcement-learning-with-emergent-roles?ref=search-presentations