Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2023.0322000

Co-evolving Multi-Agent Transfer Reinforcement Learning via Scenario Independent Representation

AYESHA SIDDIQUA[1], SIMING LIU[2], (Member, IEEE), AYESHA S. NIPU[3], ANTHONY HARRIS[4], and Yan Liu[5], (Member, IEEE)

[1]Department of Computer Science, Missouri State University, Springfield, MO, 65987 USA (as995s@missouristate.edu)
[2]Department of Computer Science, Missouri State University, Springfield, MO, 65987 USA (simingliu@MissouriState.edu)
[3]Department of Computer Science & Software Engineering, University of Wisconsin-Platteville, Platteville, WI, 53818 USA (nipua@uwplatt.edu)
[4]Department of Computer Science, Missouri State University, Springfield, MO, 65987 USA (anthony999@MissouriState.edu)
[5]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (liuyan@hnu.edu.cn)

Corresponding author: Ayesha Siddiqua (e-mail: as995s@missouristate.edu).

ABSTRACT Multi-Agent Reinforcement Learning (MARL) is extensively utilized for addressing intricate tasks that involve cooperation and competition among agents in Multi-Agent Systems (MAS). However, learning such tasks from scratch is challenging and often unfeasible, especially for MASs with a large number of agents. Hence, leveraging knowledge from prior experiences can effectively expedite the MARL learning process. Prior work has shown that we successfully facilitated transfer learning for MARL by consolidating various state spaces into fixed-size inputs, enabling a single unified deep-learning policy applicable to several scenarios within the StarCraft Multi-Agent Challenge (SMAC) environment. In this study, we expand SMAC to Multi-Player enabled SMAC (MP-SMAC) by enabling the dynamic selection of training opponents and introducing a co-evolving MARL framework, which creates a co-evolutionary arena where multiple policies learn simultaneously. Our arena comprised the simultaneous training of multiple policies in diverse scenarios, pitting them against both static AI opponents and their peers within MP-SMAC. Furthermore, we integrate co-evolution with curriculum transfer learning into Co-MACTRL framework, enabling our MARL policies to systematically acquire knowledge and skills across predetermined scenarios organized by varying difficulty levels, including evolving opponents. The results revealed significant enhancements in MARL learning performance, demonstrating the advantage of leveraging the co-evolving opponents and maneuvering skills obtained from different scenarios. Additionally, the Co-MACTRL learners consistently attained high performance across a range of SMAC scenarios, showcasing the robustness and generalizability of Co-MACTRL.

INDEX TERMS Deep reinforcement learning, multi-agent system, transfer learning, curriculum learning, co-evolutionary multi-agent reinforcement learning, StarCraft II, SMAC

I. INTRODUCTION

REMARKABLE accomplishments have been achieved in the field of Artificial Intelligence (AI) over the past decades. Gaming platforms, exemplified by Atari games [1], board games [2], [3], poker [4], and driving simulations [5], have served as invaluable test beds for AI exploration. The confined physics and finite action space within single-agent environments in these games present decision-making challenges similar to those found in real-world problems, rendering them ideal platforms for AI research. However, many real-world problems characterized by complex rules and the involvement of multiple agents pose more significant challenges for AI research. Given the presence of diverse agents in both cooperative and competitive settings, our goal is to explore AI techniques specifically designed for Multi-Agent Systems (MAS), extending beyond the domain of single-agent systems.

Among various AI approaches, reinforcement learning (RL) combined with deep neural networks (DNN) has achieved significant breakthroughs for addressing real-world problems [1], [6], [7]. The framework of deep reinforcement learning (DRL) presents a promising approach for enabling

intelligent agents to learn end-to-end solutions for complex tasks, performing at levels comparable to human champions across various domains. Techniques like the Deep Q-Network leverage experience replay and target networks to stabilize the training process by reducing sample correlation [1]. For instance, AlphaGo [8], a computer program that defeated the world champion on the board game *Go*, employs the policy gradient method of DRL. However, extending single-agent RL to Multi-Agent Reinforcement Learning (MARL) for solving MAS problems presents a significant challenge due to the constrained generalization capacity of traditional RL algorithms. The core of this issue lies in the exponential expansion of states with an escalating number of agents. To address these challenges and achieve collective objectives, a popular MARL learning paradigm called centralized training and decentralized execution (CTDE) has emerged, where complete information is employed during training and only local observations are utilized during execution. In this paper, we adapt the Centralized Training with Semi-Centralized Execution (CTSCE) learning paradigm which is introduced in our prior work [9]. This approach involves utilizing complete information during the training phase and abstracted global information during execution, assisting learning policies in discovering team objectives.

While DRL and MARL have achieved significant success in many domains, their effectiveness in tackling complex problems in large-scale multi-agent settings is constrained by the extensive training data required and prolonged learning periods. To tackle this challenge, extensive research has focused on transfer learning (TL) to reduce sample complexity and accelerate the learning process for autonomous agents handling complex learning tasks in MAS. TL leverages knowledge learned from previous tasks or external resources like human demonstrations or guidance from other learning agents to expedite learning. To create flexible and robust techniques for autonomously leveraging knowledge, TL has made remarkable progress for RL in complex applications. However, the advancement of TL for MARL still requires further development to reach real-world applications and attain efficient autonomous learning capabilities.

Considerable efforts have been invested in crafting specialized neural networks (NN) and thorough training methodologies to facilitate TL within multi-agent environments [10]. In such scenarios, the crucial aspect lies in integrating multi-modal data encoding and decoding to facilitate knowledge transfer among agents and enable curriculum learning is crucial in enhancing MARL performance. One example of enabling TL for MADDPG was done by Zhang et al. [11], focusing on training autonomous controllers in multi-UAV combat. Their study correlates the input dimension of NN with the number of agents, representing the agents' observation of the entire environment. However, this method is applicable solely when the number of agents remains constant during transfer training. The challenge of enabling autonomous agents capable of expedited learning through the reuse of knowledge from diverse sources in MAS remains unresolved.

To address the aforementioned challenges and overcome the existing obstacles, in our prior work [12], we incorporated a spatial feature encoding technique that unifies the individual agents' state inputs to NN and presents a standardized output representation, regardless of different multi-agent scenarios. We use an influence map (IM) as a spatial abstraction technique to consolidate various local observations into a consistent dimension combined with abstracted global information from the multi-agent IM (MAIM) [13], empowering agents to achieve scenario-independent capability. The spatial feature representation is combined with an agent's present state, training fixed-size NN policies that retain domain knowledge across various scenarios in MAS.

In our previous experiments, we evaluated the performance of our MARL algorithms using the standard StarCraft Multi-Agent Challenge (SMAC), where our MARL agents train against the built-in StarCraft AI. Although the built-in AI poses a formidable challenge, it constrains the potential for ongoing evolution in the learning agents and limits our MARL policies from further improvement once it has mastered defeating the static AI opponents. To overcome this constraint and facilitate ongoing improvement for MARL agents, we extended the standard SMAC to multiplayer SMAC (MP-SMAC) to accommodate dynamic AI as training opponents and evaluators, transforming MP-SMAC into an optimal co-evolutionary MARL platform. This not only supports the evolution of opponents but also allows multiple MARL policies to concurrently engage and evolve in learning. We then conducted a thorough analysis to assess our TL model's performance across different SMAC and MP-SMAC scenarios. Our approach exhibits promising results, demonstrating robustness and scalability in both intra-agent and inter-agent knowledge transfer among agents. The primary contributions of this research can be summarized as follows:

1) We expanded SMAC into multi-player enabled SMAC and created adversarial scenarios where both sides can be controlled by learnable MARL policies. Through MP-SMAC, we facilitated MARL learners to engage in competitive interactions while simultaneously learning alongside evolving opponents.
2) We introduced a curriculum transfer learning (CTL) MARL framework (MACTRL), with increasing difficulty levels incorporating both homogeneous and complex heterogeneous map scenarios. The MACTRL learners showed significant performance improvement across all the scenarios compared to baseline MARL learners.
3) Finally, We proposed a co-evolutionary MACTRL (Co-MACTRL) where multiple MARL learners engage in competition against both static AI opponents and evolving peers in a learning arena. Co-MACTRL learners developed distinct winning strategies, achieving higher win rates through competition with peers utilizing knowledge from prior scenarios.

We evaluated Co-MACTRL on MP-SMAC by training

multiple MARL policies concurrently in various scenarios, pitting them against both static AI opponents and each other within a multi-player enabled SMAC environment. Co-MACTRL utilizes scenario-independent representations to facilitate knowledge transfer among agents, leading to high MARL learning performance in complex and diverse scenarios. Furthermore, it allows our MARL policies to systematically gain expertise across predetermined learning scenarios of differing difficulties, including adapting to evolving opponents. The results revealed significant enhancements in multi-agent learning performance, demonstrating the advantage of leveraging maneuvering skills obtained from different scenarios compared to agents learning from scratch.

Our MARL algorithm shows great potential for applications in real-world multi-agent systems, particularly in facilitating information sharing and fostering cooperation among agents. For instance, a group of combat aircraft could progressively improve their maneuvering abilities and collaborative fighting tactics through the implementation of our MACTRL algorithm. Furthermore, the Co-MARL approach can be applied to the training of autonomous vehicles, where learners alternate between training and simulating adversarial roles, enhancing adaptability and robustness in varying conditions. This approach enhances sample efficiency, thereby improving the overall learning performance of the vehicle.

The rest of this paper is organized as follows. In Section II, we delve into the existing research in MARL, TL, and co-evolutionary learning in MAS. Section III details the approaches taken in our experimentation. Following this, Section IV presents the outcomes of the experiments, and Section V summarizes the findings and proposes potential future directions for this work.

II. RELATED WORK

Numerous studies have utilized MARL to train agents to achieve collective objectives in MAS environments. Early works on RL methods centered around single-agent domains. Watkins and Dayan proposed Q-Learning for agents to act optimally in single-agent Markovian problems [14]. Konda et al. introduced the Actor-Critic (AC) RL algorithm which combines the advantages of both Q-Learning and policy gradient to further enhance the RL learning performance [15]. Later, Schulman et al. introduced Proximal Policy Optimization (PPO), demonstrating its effectiveness in reducing result variance [16]. Building upon PPO, Yu et al. extended it to MAPPO, specifically tailored for multi-agent scenarios [17].

While DRL and MARL have seen tremendous success across various domains, their capability to address intricate issues within large-scale multi-agent settings is hindered by the substantial training data needed and prolonged learning duration. To address this challenge, considerable research works have concentrated on TL to diminish sample complexity and expedite the learning curve for autonomous agents managing intricate learning tasks within MAS. Most researchers presume that the knowledge acquired during the training phase remains consistent, which may not hold true in real-world applications where agents might lack perfect knowledge of the task. In such cases, despite carrying knowledge from previous tasks, agents still need to explore the environment to learn the optimal policy. Koga et al. proposed the aggregation of learned policies into a unified abstracted representation for effective performance in multi-agent scenarios [18]. However, their approach fails to choose the optimal policy, especially with a large observation space leading to scalability issues that require further investigation of feature extraction before policies can be effectively generalized. The careful selection of crucial features in TL holds significance as arbitrary information may produce subpar performance due to wrong learning bias. Taylor et al. supported this viewpoint, asserting that identifying the optimal knowledge to transfer relies on the standard metrics tuned for the training phase [19]. Subsequently, Jason et al. introduced a methodology to assess the transferability of features at individual layers within a neural network, providing insights into the degree of generalization [20]. Chen et al. suggested the Net2Net technique for knowledge transfer from previous networks by utilizing weighted values in the input [21]. While Net2Net focuses on function-preserving transformations between network specifications, our approach involves manipulating the input and output states without altering the neural network structures to enable knowledge transfer across neural networks.

Xu et al. introduced a novel approach utilizing Graph Neural Network (GNN) to represent the input state of RL algorithms for multi-agent combat problems [22]. Furthermore, Khan et al. developed a neural architecture based on transformers for global state representation and build order prediction, addressing the biases associated with Recurrent Neural Networks (RNN) and showcasing the superiority of transformers with positional encoding input for the decoder [23]. Despite its exceptional performance, this approach struggles with parallel loading due to constraints associated with the utilized dataset. Tan et al. introduced the Transitive Transfer Learning (TTL) framework that finds suitable intermediate domains for transferring knowledge between them. Here, the effectiveness of knowledge transfer is influenced by domain difficulty and distance [24]. In the domain of transfer learning, Liu et al. [25] presented an abstract forward model named Thought Game (TG) that beat the cheating level-10 AI in StarCraft by 90%. In contrast to this work, we focused on knowledge transferring on similar problems within the same domain.

Shao et al. [26] proposed a gradient-based SARSA algorithm where the inputs of neural networks are determined by the agent's current hitpoints, cooldown, and cumulative distance of own units in the StarCraft micromanagement system. While our research objective shares similarities with Shao's work, our approach is distinct in that we specifically considered both local and abstracted global information in the state space and specific move actions in the action space. The agent's previous and current state information is also integrated with the uniform state representation to the multi-agent training process for fine-tuned decision-making.

Recently, competitive co-evolution has been employed to enhance the performance of a group of learners by fostering competition. Rosin et al. [27] have introduced innovative approaches for the co-evolution of a population, emphasizing direct competition and evolution. Whiteson [28] demonstrated the effectiveness of evolutionary algorithms in discovering high-performance RL policies, particularly in gaming environments where competitive co-evolution leads to the simultaneous evolution of strong players and their corresponding opponents. While most research in competitive co-evolution focuses on specific evolutionary algorithms such as genetic algorithms [29], [30], our study proposes a co-evolutionary learning framework without relying on any particular evolutionary algorithm. We introduce RL as our fundamental learning algorithm for evolving and optimizing policies within a carefully crafted reward system in competitive multi-agent systems. Cotton et al. [31] have proposed a co-evolutionary RL technique, where a group of learning agents undergoes training in competitive scenarios. The most successful individuals are chosen to serve as parents for the subsequent epoch. In our Co-MACTRL approach, we have established an arena where learners evolve through direct competition with each other without implementing a selection process. Instead, we have integrated CTL with competitive co-evolution, ensuring that each learning agent receives an equal opportunity to train and evolve over time.

Olsen et al. [32] demonstrated that using RL to facilitate co-evolution between predators and prey enhanced the learning process for both parties. Pinto et al. [33] proposed a robust adversarial RL approach, demonstrating significant performance improvement in the RL learning process by including dynamic adversary agents. Additionally, Szubert et al. [34] underscored the significance of behavioral diversity and challenging opponents in driving performance gains in co-evolving agents. Drawing on their research, we enabled co-evolution among multiple training agents by pitting them against one another in the domain of MARL which showcased significant performance improvement compared to only training against the static AI opponent. Through our Co-MACTRL learning framework, we establish an environment where agents serve as opponents to one another during training, alongside the presence of a static AI opponent. This setup proposes a combination of diverse and dynamically challenging opponents for the learning agents, aiming to further enhance their training efficacy.

III. METHODOLOGY

We formulate the SMAC scenarios as Markov games, which extend the framework of Markov Decision Processes (MDP) in a multi-agent setting [35]–[37]. A Markov game comprises a collection of states representing the status of both the agents and the environment. Each participating agent has a set of actions $(A_1, A_2, ..., A_N)$ and observations $(O_1, O_2, ..., O_N)$, where N signifies the number of agents in a given episode. The ally and enemy units, along with the environmental information, are modeled as observations for individual agents to make decisions and take actions. In a Markov game, each agent follows a policy π at every step within the environment and collectively earns a shared reward r_{shared}.

$$\pi : S_{\{O_1,O_2,...O_n\}} \times \{A_1, ..., A_N\} \rightarrow S', r_{shared} \quad (1)$$

Equation 1 signifies a Markov game transition from state $S_{\{O_1,O_2,...O_n\}}$ to S', which we utilized to model our MAS where the state is formulated using observations of the agents.

A. SIMULATION ENVIRONMENT

This research builds upon our earlier work [13], [38], where we evaluated the performance of the proposed MARL models across a range of multi-agent challenge scenarios within SMAC. We continue to use SMAC as our primary research platform for all experiments that evaluate both co-evolutionary multi-agent learning and curriculum transfer learning performance. SMAC is built upon the foundation of the StarCraft II Learning Environment, as detailed in the work by Vinyals et al. [39], offering a variety of multi-agent micromanagement challenging scenarios where the goal is to eliminate opponents using a given set of units.

In our previous work, we evaluated our MARL algorithms using the standard SMAC environments, where our agents train against the built-in StarCraft AI. Although the built-in StarCraft AI poses a formidable challenge, it constrains the potential for ongoing evolution in the learning agents and limits our MARL learner from further improvement once it has mastered defeating the static AI opponents. To overcome this limitation and facilitate ongoing improvement for MARL learners, we extended SMAC to Multi player enabled SMAC (MP-SMAC) to accommodate dynamic AI as training opponents, transforming it into a co-evolutionary MARL platform. This not only supports the evolution of opponents but also allows multiple MARL learners to concurrently engage and evolve. To introduce dynamic opponents on MP-SMAC, we designed several multiplayer maps that incorporate a mix of unit types and quantities through the StarCraft II Editor. Table 1 provides an overview of the maps on SMAC and MP-SMAC utilized in this research.

TABLE 1: Multi-Agent Scenarios on SMAC and MP-SMAC

Scenario	Platform	Units on Each Side	Type
$3m$	SMAC		Homogeneous
$8m$	SMAC		Homogeneous
$2s3z$	SMAC		Heterogeneous
$3s5z\_mp$	MP-SMAC		Heterogeneous

On both SMAC and MP-SMAC scenarios, we explored two types of observation spaces for agents to create a scenario-independent representation of the state:

1) Local observation: This includes individual details for each agent, such as hitpoints, unit type, relative positions, and distances of allied and enemy agents within the observation range.

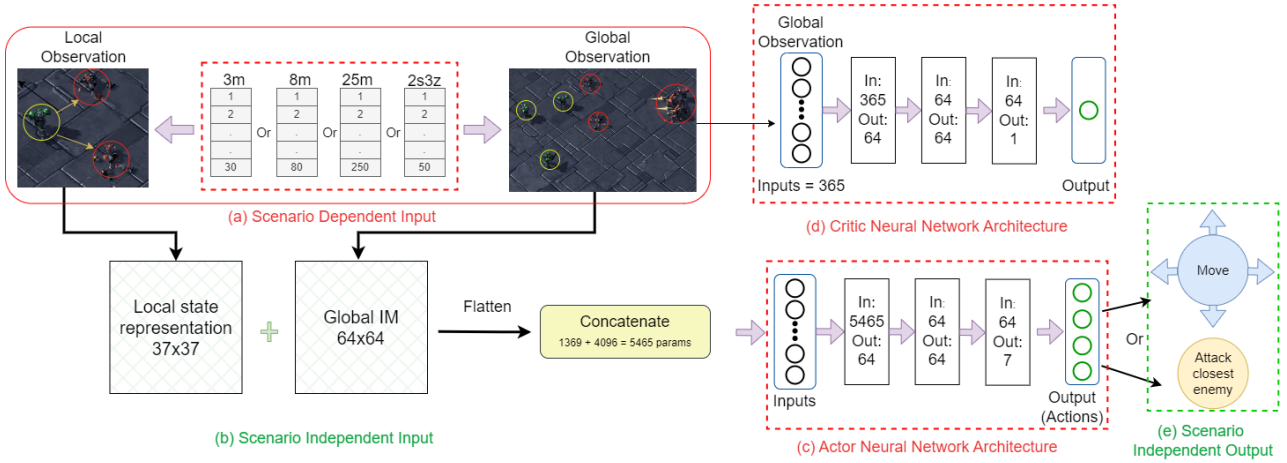


FIGURE 1: Transfer learning model representation for single agents. The agents' observation is abstracted in fixed size of influence maps and the action space is also fixed into a finite set, including movements and attacking the closest enemy.

2) Shared global abstraction: Here, all agents on the map are aggregated along with their features derived from local observations. This abstraction also includes additional details like weapon cooldown and previous actions of each agent.

Both the local and abstracted global observations' features are normalized within the range $[0, 1]$ in our experiments and utilized as input space to our MARL algorithms in a unified manner.

B. CURRICULUM TRANSFER LEARNING ARCHITECTURE

To speed up the MARL learning process, we previously introduced a novel Curriculum Transfer Learning (CTL) framework which utilizes a scenario independent state and action representation to preserve and reuse knowledge across the scenarios. In SMAC, states are typically depicted based on the number of units in each scenario, as illustrated in Fig. 1a. For scenarios $3m$, $8m$, and $25m$, the size of the local state is provided in the form of 30, 80, and 250 one-dimensional vectors respectively which are marked with a dotted rectangle of red color in Fig. 1a. The default state representation is compact and carries precise agent information for training MARL. However, it limits the knowledge transfer over multiple scenarios due to the scenario-dependent state size. To mitigate this problem, in our prior work, we presented a fixed-size state representation that is unified in such a way that the state size remains constant, irrespective of the number of agents in the scenario. In this study, we utilized the unified representation with an improved MARL architecture to experiment with intricate map scenarios in both SMAC and MP-SMAC. The subsequent sections delve into the specifics of the MARL components utilized in our experiments.

1) Scenario Independent State Representation

To create a unified input state representation, we considered both agents' local observations and abstracted global information. In our prior work, we employed a spatial information technique called Agent Influence Map (AIM) to extract and filter aggregated spatial representation from the global information in order to discover common objectives and promote the learning of collaborative behaviors among agents. We further extended the use of AIM in this study to construct a scenario-independent local state representation for enabling knowledge transfer across all scenarios provided in SMAC and MP-SMAC. Each AIM is determined by three parameters: the current relative health of the agent I_0, the influence decay rate which equals the inverse of the distance from the agent λ_I, and the range of influence d_I. A negative weight is used for enemy agents in order to differentiate them from the allied agents. With an aggregation of all the agents' AIM, a generalized and more robust Multi-Agent Influence Map (MAIM) is formed. Based on the performance of different dimensions, a 64×64 MAIM representation has been used for unifying the abstracted global information in further experimentation.

In our state representation of local observation, we considered the local observations of the prior step, the actions performed in that step, along with the current step information. The local observations include distance, relative position, health, shield, and unit type for allied and enemy units within the sight range of each agent. The default states received from SMAC depend on the number of active agents in the game environment. In order to remove the dependency on the number of agents across SMAC scenarios, we extended the use of IM from global information abstraction to local observation aggregation. The local IM transformation yields a fixed dimension of sight range with a resolution of 37×37, as determined by experimental findings. The unified subset of global and local information as shown in Fig. 1b is then flattened and propagated through the neural network, which applies to all SMAC scenarios.

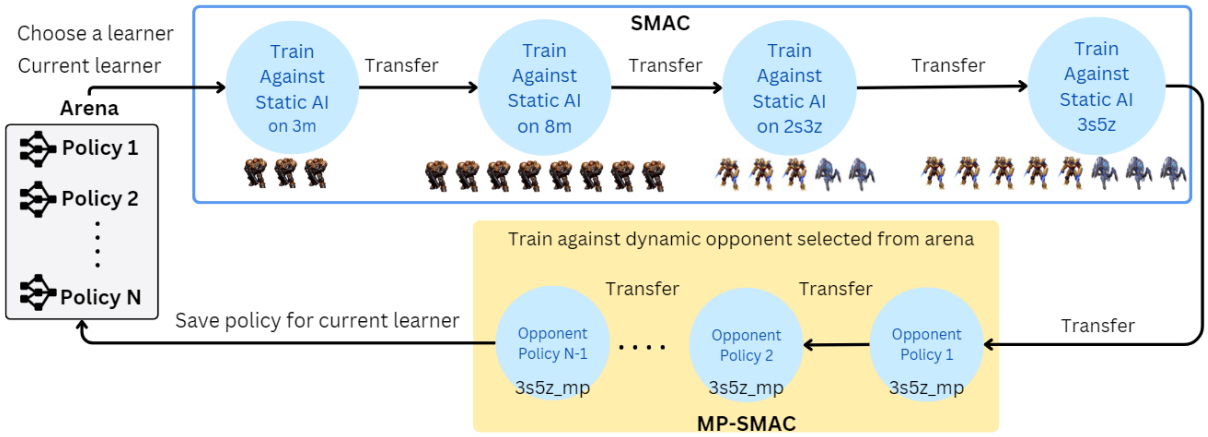


FIGURE 2: Co-MACTRL framework for N independent learners. Each learner takes turns following a predefined CTR step in standard SMAC and then competes against peers in MP-SMAC to evolve concurrently.

2) Multi-Layer Perceptron (MLP) Structure

To conduct our experiments, we utilized the Asynchronous Advantage Actor-Critic (A3C) architecture as the foundational learning algorithm. Minh et al. [40] introduced the A3C framework, which stands out as state-of-the-art across various gaming tasks. This A3C algorithm offers the flexibility of choosing complete information and local observation, enabling seamless integration into both the training and execution processes, aligns well with our Centralized Training with Semi-Centralized Execution (CTSCE) learning approach, as well as other variations. In our A3C configuration, we utilize separate MLP components for the actor and critic networks, without sharing neural layers between them. Each MLP consists of an input layer defined by the state space, two fully connected hidden layers each with 64 neurons, and an output layer determined by the unified action space. As illustrated in Fig. 1c, we employ multiple agents with a shared single neural network under a multi-task learning scheme, hence alleviating the computational burden during both training and inference, as only one network requires evaluation. This design facilitates faster learning as the parameters in the neural networks are updated concurrently for each agent. During the MARL learning process, each agent independently selects an action from a discrete action space according to the MLP policy. This is done in a decentralized manner, with the agent utilizing its scenario independent state representation as input. Subsequently, we use the critic neural network shown in Fig. 1d to evaluate the effectiveness of that action and reward each agent a shared reward based on the outcomes. While our earlier method demonstrated impressive results on less intricate maps such as $3m$ and $8m$, it exhibited reduced performance in heterogeneous scenarios like $2s3z$ and $3s5z$. All of the results are collected from 7 different random seeds with each experiment lasting for up to 80 million steps. We evaluate the performance of the trained MARL model every 25 game episodes throughout the entire training process.

3) Scenario Independent Action Representation

In SMAC, agents are required to make decisions choosing from a finite action space based on the state information. For the move actions, agents typically have four directions to choose from: north, south, east, and west. However, when it comes to attacking decisions, agents must consider the number of enemies in their current local observation within the sight range in the default action space provided in SMAC. This presents a challenge, as the number of enemies can vary greatly from scenario to scenario. To address this issue, we propose a generalized approach that only considers the closest enemy position for the attacking action to remove the dependency on the number of agents within the sight range and make attack actions without knowing specific enemy agents. This approach is illustrated in Fig. 1d, where the scenario-dependent output is replaced by an attack action following our custom-generated policy. As we trained our NNs with the spatial information of the agent's current position with detailed policies, targeting the closest enemy instead of choosing the default scenario-dependent action doesn't lose any valuable information required to take action. The generalized outcome enables the agent to share attacking policies, thereby facilitating the transfer of knowledge across a broad range of SMAC environments. By reformulating the existing solution in a unified manner, we were able to achieve both improved performance in large-scale scenarios and enhanced efficiency in the simulation environment.

C. CURRICULUM TRANSFER LEARNING PROCEDURE

Curriculum learning is a specific type of transfer learning that arranges a series of tasks according to their increasing level of complexity [41]. This approach involved training on how to play a game against simulated opponents who became progressively more competent, allowing the agents to learn useful strategies and gain knowledge that they could apply to real-world challenges [42]. In our previous experiments, we delved into the intriguing question of how the transfer of

winning strategies among *Marines* learned in scenarios $3m$ and $8m$ can positively impact the behavior of *Stalkers* and *Zealots* in an extended heterogeneous environment $2s3z$. The curricular process flow of our training policy from the simplest scenario, $3m$, retrains the learned model in a medium-level scenario, $8m$, and finally carries the knowledge learned from $3m$ and $8m$ to tackle a much more complex scenario $2s3z$ with different unit types and heterogeneous team units that the policy has never seen in prior training scenarios. Building upon our previous study, we continued the transfer learning curriculum with scenario $3s5z$, involving three *Stalkers* and five *Zealots* in competitive interactions on MP-SMAC. The CTL process is summarised in Equation 2.

$$\pi : S_{\{m_i\}} \times \{A_u\} \rightarrow S_{\{m_{i+1}\}} \times \{A_u\} \rightarrow ... \rightarrow S_{\{m_N\}} \times \{A_u\}$$
$$C \in \{m_1, m_2, ...m_N\}, i \in \{1, 2, ...N\} \tag{2}$$

We characterize curriculum C as a compilation of SMAC Maps $(m_1, m_2, ..., m_N)$, each presenting escalating challenges and encompassing diverse combinations of ally and enemy units. Initially, the MARL policy π is trained within the start map m_1 using a unified state S_m and action A_u representation independent of specific scenarios. Subsequently, the policy undergoes training across successive tasks within curriculum C until reaching the designated final map m_N. A comprehensive detail of the CTL process is provided in Algorithm 1.

Algorithm 1 MACTRL Algorithm

```
procedure MACTRL(maps[], policy, curriculum_len)
    for map in maps[] do
        env ← load env for map
        buffer ← init Replay Buffer for map
        for stepCount ← 1 to curriculum_len do
            take_actions(policy)
            step + +
            update_policy(policy)
        end for
    end for
    return policy
end procedure
```

Additionally, we introduced novel evaluation metrics to appraise the robustness and adaptability of the CTL learner across diverse SMAC scenarios. We further extended our MARL learning process to utilize the effectiveness of CTL and co-evolution and propose a dynamic co-evolving learning framework Co-MACTRL. The details of the Co-MACTRL framework are described in the following subsection.

D. CO-EVOLVING LEARNING FRAMEWORK FOR MULTIPLE LEARNERS

Inspired by the natural world, co-evolution leverages competitive forces to foster the development of more effective behaviors [31]. In this research, we introduced Co-MACTRL that facilitates the evolution of multiple learners through competition against both stationary AI opponents and fellow learners across a variety of SMAC and MP-SMAC maps, leveraging curriculum transfer learning.

Algorithm 2 Co-MACTRL Algorithm

```
maps[] ← [3m, 8m, 2s3z, 3s5z]
arena[] ← [Learner1, Learner2, Learner3]
step ← 0
while step ≤ max_step do
    for curLearner in arena[] do
        curPolicy ← MACTRL(maps, curPolicy, 1M)
        for enemyPlayer in arena[] do
            if curLearner ≠ enemyPlayer then
                enemyPolicy ← init policy for 3s5z_mp
                env_mp ← load env for 3s5z_mp
                if policy saved for enemyPlayer then
                    enemyPolicy ← load saved policy
                end if
                for stepCount ← 1 to 1,000,000 do
                    take_actions(curPolicy, enemyPolicy)
                    step + +
                    update_policy(curPolicy)
                end for
                save_policy(curPolicy)
            end if
        end for
    end for
end while
```

The detailed Co-MACRTL framework is shown in Algorithm 2, where at first, we established a CTL sequence that includes both homogeneous and heterogeneous map scenarios organized by difficulty levels. For our experiments, we use CTL in the sequence of $3m \rightarrow 8m \rightarrow 2s3z \rightarrow 3s5z$, fostering inter-agent and intra-agent knowledge transfer. Given the substantial computational load, we introduce three separate MARL learners in a co-evolutionary arena, enabling them to train and compete simultaneously against stationary AI opponents and each other. Each MARL learner undergoes sequential training for 1M steps against the built-in StarCraft AI across maps such as $3m$, $8m$, $2s3z$, and $3s5z$. Note that the 1M game steps were selected based on experimental results after trying several different training lengths on CTL maps.

After completing 4M game steps against the built-in AIs on CTL maps, the learner begins selecting a fellow learner from the arena as the opponent and engages in gameplay against this dynamic opponent within our MP-SMAC environment. Given that the 3 *Stalkers* and 5 *Zealots* scenario presents the most difficult challenge in our CTL sequence, we exclusively train our MARL learners against the dynamic opponent using only $3s5z\_mp$. Additionally, training against dynamic opponents across all scenarios would introduce substantial computational complexity and significantly prolong our training duration. To streamline the collection of Co-MACTRL training outcomes, we opt to focus solely on the most intricate map scenario for co-evolutionary learning. Engaging

with constantly improving opponents offers the chance to surpass the constraints of the static StarCraft AI and develop novel winning strategies through enhanced skill and collaboration. Within the learning arena, each MARL participant takes turns following this learning procedure and the evolving cycle persists until the maximum training length is reached. Within this co-evolutionary MARL framework, we utilized the advantages of scenario-independent state representation and unified action space, and seamlessly integrated these elements with CTL and our MP-SMAC environment, fostering the evolution of multiple learners over time. The learning structure depicted in Fig. 2 is the Co-MACTRL learning architecture. In this framework, several learners go through a CTL sequence before engaging in competition with adaptive adversaries within the MP-SMAC environment, facilitating a process of co-evolution.

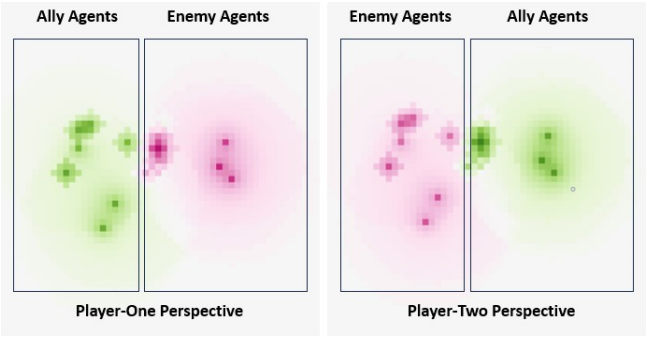


FIGURE 3: Multi Agent Influence Map for two players SMAC

To match the input space of the opponent policy, we created MAIM from the enemy agent's perspective and crafted unified local and global observations to guide the decision-making of peer learners acting as opponents. In order to differentiate between allied and enemy agents, the initial influence of allied units is the negative value of their relative health as provided by SMAC. Despite sharing the same global observation, we maintained two separate MAIMs to represent the perspectives of the two adversarial players. An example of multi-agent influence maps from two players' perspectives is shown in Fig 3. The maps are derived from the same game step where the green color denotes the ally team's influence and the purple color denotes the opponent team's influence.

In our earlier research, we discovered that MARL learners can leverage insights acquired from simpler scenarios to gain proficiency in more complex ones, employing curriculum transfer learning. In this study, we propose a co-evolutionary MARL framework for a group of concurrent learners, where each learner trains through direct competition with others within a fixed arena of learners. Furthermore, by incorporating training phases in a cyclic fashion that integrates CTL with co-evolutionary learning, the MARL learners within the arena progressively enhance their capabilities with each cycle. In contrast to our prior CTL approach, where a learner concludes its curriculum learning without revisiting it, our hypothesis suggests that repeating curriculum transfer learning in cycles can enable learners to develop a more robust learning technique.

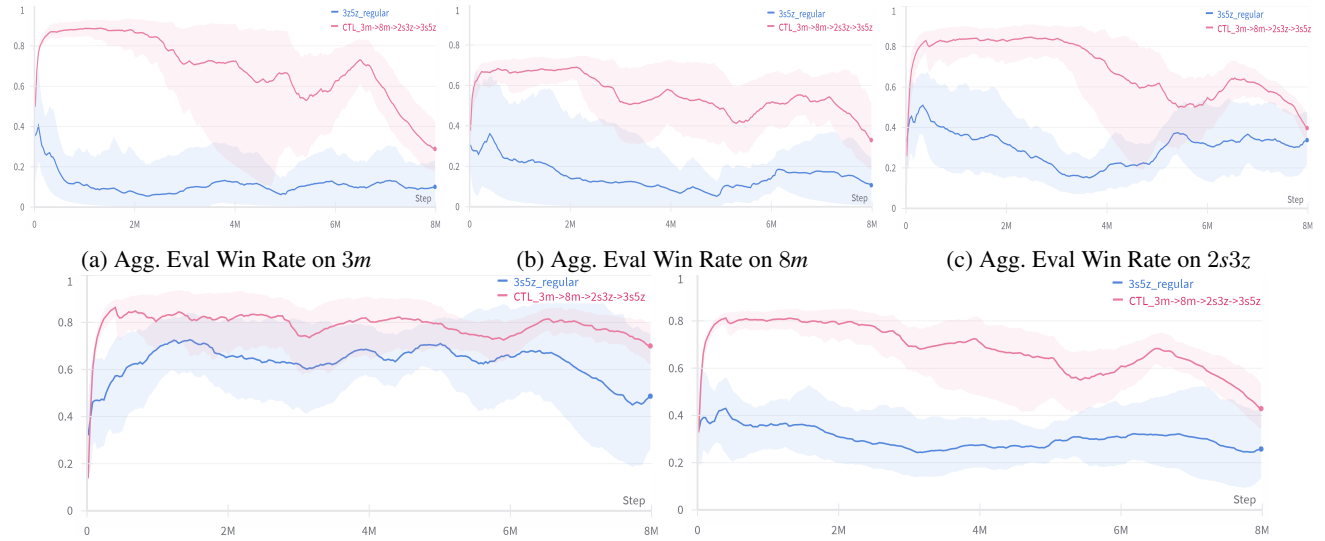
IV. RESULTS AND DISCUSSION

Co-MACTRL's transfer learning presents a promising solution to enhance agents' asymptotic performance, empowering them to achieve higher proficiency in mastering intricate tasks in MAS. To facilitate knowledge transfer, we adapted the scenario-independent input and output representations that allow one unified deep-learning policy viable across various scenarios within SMAC. In our earlier experiments, we compared performance across different input state resolutions with varying local state dimensions ranging from 19×19 to 55×55. As experimental results demonstrated that, 37×37 yielded the highest performance, we adapted it as the local state representation for our subsequent experiments. The performance of our Co-MACTRL has been evaluated across both homogeneous and heterogeneous scenarios in SMAC and MP-SMAC. This evaluation involved multiple performance metrics, including average winning rates within specific scenarios and an overall performance evaluation calculated across all 7 instances of each co-evolutionary training.

A. CURRICULUM TRANSFER LEARNING FOR MARL

MACTRL's curriculum transfer learning demonstrates substantial performance enhancements across both homogeneous and heterogeneous scenarios. In our prior work, we relied on average episode rewards to evaluate learning performance. To delve deeper into MACTRL's and Co-MACTRL's learning capabilities, we've implemented a comprehensive evaluation matrix across all experiments conducted in this study. We arranged CTL in the sequence of $3m \rightarrow 8m \rightarrow 2s3z \rightarrow 3s5z$ ordered by the difficulty levels, aiming to investigate how knowledge is acquired and transferred throughout the learning process. In our CTL procedure, a MARL policy undergoes training against static AI in the $3m$ scenario for 1M game steps, following which it learns to navigate the $8m$ scenario for an additional 1M steps. Sequentially, we move on to the $2s3z$ scenario and, finally, the $3s5z$ scenario. Our goal is to understand how much knowledge the learning MARL policy retains from the $3m$ scenario while being trained in the $8m$ scenario at the early training stage. Similarly, as the training advances towards more challenging scenarios like $2s3z$ and $3s5z$, we anticipate that the knowledge and skills acquired from previous scenarios could potentially augment the learning performance or expedite the learning process. We hypothesize that training our MARL policy across all four maps following the CTL sequence would not only achieve high performance quickly but also foster generalized playing skills rather than specialized ones for specific scenarios.

To demonstrate the CTL policy's robustness and generalizability, we designed a comprehensive evaluation matrix encompassing all four maps outlined in the CTL sequence. At each evaluation phase, we conducted 32 distinct game episodes on each map separately, enabling us to analyze the



(a) Agg. Eval Win Rate on $3m$ (b) Agg. Eval Win Rate on $8m$ (c) Agg. Eval Win Rate on $2s3z$

(d) Agg. Eval Win Rate on $3s5z$ (e) Agg. Average Win Rate over 4 Scenarios

FIGURE 4: Results of MACTRL on $3m \rightarrow 8m \rightarrow 2s3z \rightarrow 3s5z$ compared to MARL only on $3s5z$.

TABLE 2: Peak evaluation results against built-in AI opponent during 8M training steps across scenarios.

MARL Algorithms	**Avg Win Rate**	$3m$	$8m$	$2s3z$	$3s5z$
MACTRL Learner	**80%**	**87%**	**68%**	**84%**	**85%**
MARL Leaner $3s5z$	40%	38%	41%	50%	72%

average win rate for each map individually and determine the overall average win rate of allied agents. Fig. 4 displays the CTL results obtained from seven different random seeds across 8M training steps. The blue lines represent training outcomes for the $3s5z$ scenario without prior knowledge, while the red lines depict outcomes for CTL following the sequence $3m \rightarrow 8m \rightarrow 2s3z \rightarrow 3s5z$. The MACTRL learner initiated training in a homogeneous scenario, $3m$, for 1M steps from scratch, achieving an 87% winning rate during this training phase, as illustrated in Fig. 4a. In contrast, the standard $3s5z$ learner exhibited subpar performance in the $3m$ scenario, attaining an average winning rate of only 15%.

Following completion of the training on $3m$, the MACTRL learner progressed to training against the $8m$ scenario, which presents a more challenging homogeneous scenario featuring 8 *Marines*. The evaluation depicted in Fig. 4b shows that the learner equipped with prior knowledge from the $3m$ scenario exhibited a rapid learning curve, achieving a 68% winning rate against the built-in AI opponent in the $8m$ scenario. In contrast, the regular learner shows a poor performance on the $8m$ scenario by achieving only a 20% winning rate. Notably, the MACTRL learner's performance on the $3m$ map scenario remained consistent even when not actively training in that specific scenario, suggesting the retention of prior knowledge while acquiring expertise in the new scenario.

After finishing the training on $8m$, the learner proceeded to train in the heterogeneous scenario, $2s3z$. The evaluation results depicted in Fig. 4c show the exceptional performance of the MACTRL learners, achieving an 85% winning rate. On the contrary, the learner exclusively trained on $3s5z$ achieved only a 30% winning rate in the $2s3z$ scenario. Throughout our assessments on the $3m$, $8m$, and $2s3z$ maps, the regular learner exhibited subpar performance, as these scenarios were entirely unfamiliar to the learner whereas the MACTRL learner consistently demonstrated improved performance, retaining its proficiency in the simpler scenarios. Finally, we extended our CTL to the complex heterogeneous scenario, $3s5z$, and trained the MACTRL learner for another 5M steps. Leveraging prior knowledge from simpler map scenarios, the learners achieved an 85% winning rate within the initial training steps. In contrast, the regular MARL learner archives a 72% winning rate within the entire 8M steps. The evaluation result on $3s5z$ is presented in Fig. 4d, demonstrating that the MACTRL learner, equipped with prior knowledge, achieves excellent performance faster than regular learners.

The highest average win rate achieved by MACTRL learners across the four scenarios is 80%, showing the robustness and generalizability of the CTL learning approach in addressing various MARL scenarios. The maximum training values displayed in Table 2 demonstrate that the MACTRL learner outperforms regular MARL learner by an average of 50% across all four scenarios. Fig. 4e shows the average winning rate of the learners at each evaluation phase across all four scenarios. The MACTRL learners maintained an 80% winning rate on average during the first 4M training steps. Based on the training outcomes, we can assert that prior knowledge in the simpler map facilitates the MACTRL learners in achieving competitive performance more rapidly than the regular learner, establishing a winning strategy applicable to all scenarios.

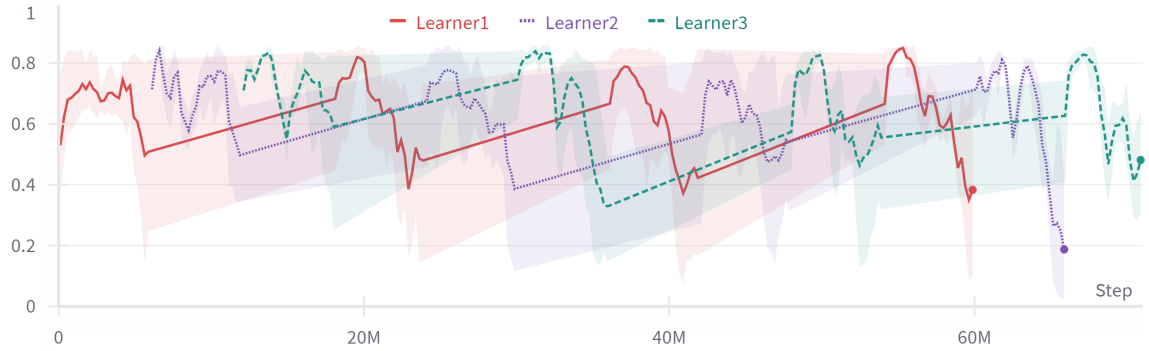


FIGURE 5: The aggregated average win rates from evaluations involving three MARL learners in Co-MACTRL on MP-SMAC scenarios $3m$, $8m$, $2s3z$, and $3s5z$ when competed against the standard built-in AI opponents.

TABLE 3: Peak evaluation results over cycles of Co-MACTRL and Co-MARL learners.

Learner	Scenario	Cycle 1		Cycle 2		Cycle 3		Cycle 4	
		Co-MACTRL	Co-MARL	Co-MACTRL	Co-MARL	Co-MACTRL	Co-MARL	Co-MACTRL	Co-MARL
Learner1	$3m$	80%	55%	85%	42%	82%	32%	**90%**	38%
	$8m$	60%	52%	68%	29%	62%	26%	**73%**	29%
	$2s3z$	89%	59%	87%	63%	85%	60%	**90%**	54%
	$3s5z$	83%	65%	89%	67%	86%	73%	**90%**	65%
Learner1	avg. win rate	75%	53%	82%	50%	78%	44%	**84%**	39%
Learner2	avg. win rate	**80%**	44%	78%	45%	73%	45%	79%	38%
Learner3	avg. win rate	83%	51%	83%	42%	82%	38%	**83%**	54%

However, as the MACTRL learner continues training on $3s5z$, its performance on the earlier maps gradually declines. This trend is evident in the results depicted in Fig. 4, where after 7M steps, the performance on $3s5z$ stabilizes, but the performance on the other three maps starts to decline. To avoid the deterioration of the performance and to enable multiple learners to train and evolve together within a co-evolving arena following a CTL sequence in an iterative manner, we have introduced the Co-MACTLR framework described in Subsection III-D. The results of the Co-MACTRL are described in the following subsection.

B. CO-MACTRL PERFORMANCE

Given the substantial computational load, we select three learners arbitrarily to assess the co-evolutionary performance among the learning participants in our Co-MACTRL learning experiments. The three individual MARL learners take turns to train and evolve against both static AI opponents as well as peer competitors across multiple SMAC and MP-SMAC scenarios. The training process operates iteratively, and the assessment results, depicted cyclically in Fig. 5, illustrate the ongoing evolution of all the MARL learners. We established an arena comprising three MARL learners, designated as Learner1, Learner2, and Learner3. During the training, Learner1 engages with the built-in SMAC AI in the map sequence of $3m \rightarrow 8m \rightarrow 2s3z \rightarrow 3s5z$ for 1M training steps on each scenario. Following the initial 4M training steps, Learner1 transitions to training against the other two learners, Learner2 and Learner3, on $3s5z\_mp$, with each comprising 1M steps, totaling 6M steps for Learner1. Fig. 5 displays the aggregated average win rate across all four maps for the three learners, illustrating that Learner1 achieves a 75% average winning rate during the first training phase. The performance of the three learners in each scenario is depicted in Fig. 5.

After the completion of Learner1, Learner2 initiates the CTL training procedure, following the same curriculum learning process as Learner1. Learner2 confronts predefined SMAC maps from 7M to 10M steps mark before engaging in competition and training against Learner1 and Learner3 as opponent policies for an additional 2M steps. Note that, Learner1, having mastered the CTL maps, poses novel challenges to Learner2, whereas Learner3 is not yet trained. Learner2 attains an average winning rate of 80% during the first cycle of training. Subsequently, Learner3 follows the same curriculum as Learner1 and Learner2, completing the initial training phase in 18M training steps and achieving an average winning rate of 83% across all maps.

The Co-MACTRL learners undergo continuous training in a sequential manner, repeatedly engaging with the same curriculum. In each training phase, the learners face increasingly formidable opponents, as these opponents evolve alongside them, fostering a process of co-evolution. The highest evaluation win rates of Learner1 on all four maps throughout the training cycles are outlined in Table 3. It is evident that, with each iteration of the training phase, the peak winning rate shows improvement for each scenario. In the case of the $3m$ scenario, the winning rate rises from 80% to 90%, and for the $8m$ scenario, it increases from 60% to 73%. Furthermore, the winning rates for the $2s3z$ and $3s5z$ scenarios also exhibit gradual increments with the progression of training cycles.

Moreover, the Co-MACTRL learners surpass the MACTRL learners by a margin of nearly 7% when considering the peak evaluation values for each map scenario. This indicates that the dynamic arena framework used by Co-MACTRL learners has empowered them to discover distinctive winning strategies through engagements against evolving opponents, a capability absent in MACTRL, which relies solely on static AI opponents. Note that, the drops shown in the results are due to training against dynamic opponents but evaluating against the built-in SMAC AI. For our experiments, we considered SMAC AI as the baseline performance evaluator. Despite the declines evident at the end of each cycle, the learners exhibited performance enhancement in subsequent cycles, indicating that the dynamic opponents aid in exploring and expanding the learning paradigm, resulting in unique maneuvering skills.

To demonstrate the efficacy of Co-MACTRL, we conducted a comparison with Co-MARL, consisting of three individual learners evolved and assessed on the $3s5z$ and $3s5z\_mp$ maps. Notably, both Co-MARL and Co-MACTRL learners undergo cycles of competition and evolution. However, Co-MACTRL incorporates CTL sequences during training, whereas Co-MARL does not. The training outcomes are depicted in Table 3. The evaluations on the $3s5z$ map indicate that Co-MARL learners showed performance improvement over four training cycles, with *Learner*1 achieving a peak winning rate of 73%, comparable to Co-MACTRL learners. However, their performance significantly decreased in other maps that the learners had not encountered before, resulting in an average winning rate nearly 50% lower than that of Co-MACTRL learners. These results suggest that integrating CTL with Co-evolution effectively generates a robust and generalized winning strategy across all scenarios.

C. LEARNED BEHAVIOR ANALYSIS

After training the Co-MACTRL learners for 72M game steps over four complete cycles of each learner, we collected the best-performed learners and examined their acquired behaviors across various scenarios. The Co-MACRTL learners learned to position and attack the opponents coordinately. Remarkably, this learner exhibited outstanding performance in both simple scenarios like $3m$ and intricate ones like $3s5z$. Fig. 6 illustrates the Co-MACTRL learner's initial positioning and focus fire on a specific enemy agent as a team across four distinct scenarios. The strategy deployed by the learner is commonly referred to as "focus fire", which involves a player directing a group of units to collectively attack a designated target, aiming to eliminate enemy units more rapidly.

In Fig. 6a, all three ally *Marines* concentrate their fire on a single enemy unit highlighted by a red square box. In the $3m$ scenario, the allied units keep firing at the opponent units one by one, leading to a strong winning strategy effective in almost 90% cases. When the same strategy transfers to a larger team with 8 *Marines* shown in Fig. 6b, the allied team coordinates their attack on four distinct enemy agents. Notably, the transfer of knowledge from the simple scenario has led the learner to find the winning strategy faster as shown in Fig. 4. To further assess the effectiveness of the knowledge transferring between different types of agents (*Marine* to *Zealot* and *Stalker*), we evaluated the learner on both $2s3z$ and $3s5z$ scenarios shown in Fig. 6c and Fig. 6d respectively. Although the allied team is following the same focus fire strategy, the allied unit maintains the position of *Stalkers* and *Zealots* separately. In both scenarios, the allied units are focusing fire on the enemy unit's *Zealots* first to weaken the opponent team hence leading to a winning move.

(a) $3m$ (b) $8m$ (c) $2s3z$ (d) $3s5z$

FIGURE 6: Co-MACTRL learners performing "focus fire" on four scenarios showcasing a generalized winning strategy.

V. CONCLUSION AND FUTURE WORK

This study presented a co-evolutionary multi-agent curriculum transfer learning (Co-MACTRL) framework to enable multiple MARL policies to concurrently engage and evolve on learning cooperative tasks across various scenarios within the StarCraft Multi-Agent Challenges (SMAC) platforms. We first extended the standard SMAC platform into MP-SMAC, allowing both sides to be controlled by adaptable AI policies. Subsequently, we integrated the co-evolving MARL approach with curriculum transfer learning, empowering our MARL policies to systematically accumulate expertise across predefined learning scenarios organized by varying difficulty levels. This approach promotes knowledge transfer between agents and across various learning stages, leading to enhanced multi-agent learning performance in increasingly complex and diverse scenarios against evolving opponents. We evaluated Co-MACTRL on SMAC and MP-SMAC by simultaneously training multiple MARL policies across diverse scenarios, pitting them against both static AI opponents and peer learners within a multi-player enabled SMAC environment. Co-MACTRL employs scenario-independent representations, enabling effective knowledge transfer among agents, resulting in high MARL performance in complex scenarios. Moreover, it enables our MARL policies to systematically acquire proficiency across varying difficulty levels and adapt to evolving opponents. The results showed significant improvements in multi-agent learning, highlighting the advantage of leveraging maneuvering skills obtained from diverse scenarios over agents starting from scratch.

This study opens avenues for further exploration in several areas. Firstly, our MARL policy's testing was confined to a restricted set of scenarios, prompting future research to assess its performance across a broader spectrum of heterogeneous environments featuring more intricate maps. Moreover, there is potential to explore more advanced curricu-

lum learning designs and knowledge transfer approaches for co-evolutionary MARL, aiming to further enhance MARL learning performance. Lastly, integrating additional deep RL techniques, such as recurrent neural networks, holds promise in enhancing multi-agent systems and widening the scope for improvement.

VI. ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Award No. 2302060.

REFERENCES

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[2] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[3] Kun Shao, Dongbin Zhao, Zhentao Tang, and Yuanheng Zhu. Move prediction in gomoku using deep learning. In *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pages 292–297. IEEE, 2016.

[4] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisỳ, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 356(6337):508–513, 2017.

[5] Khan Muhammad, Amin Ullah, Jaime Lloret, Javier Del Ser, and Victor Hugo C de Albuquerque. Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4316–4336, 2020.

[6] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[7] Michael L Littman. Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521(7553):445–451, 2015.

[8] Fei-Yue Wang, Jun Jason Zhang, Xinhu Zheng, Xiao Wang, Yong Yuan, Xiaoxiao Dai, Jie Zhang, and Liuqing Yang. Where does alphago go: From church-turing thesis to alphago thesis and beyond. *IEEE/CAA Journal of Automatica Sinica*, 3(2):113–120, 2016.

[9] Paul Brackett, Siming Liu, and Yan Liu. Sc-mairl: Semi-centralized multi-agent imitation reinforcement learning. *IEEE Access*, 2023.

[10] Felipe Leno Da Silva and Anna Helena Reali Costa. A survey on transfer learning for multiagent reinforcement learning systems. *Journal of Artificial Intelligence Research*, 64:645–703, 2019.

[11] Guanyu Zhang, Yuan Li, Xinhai Xu, and Huadong Dai. Efficient training techniques for multi-agent reinforcement learning in combat tasks. *IEEE Access*, 7:109301–109310, 2019.

[12] Ayesha Siddika Nipu, Siming Liu, and Anthony Harris. Enabling multi-agent transfer reinforcement learning via scenario independent representation. In *2023 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2023.

[13] Anthony Harris and Siming Liu. Maidrl: Semi-centralized multi-agent reinforcement learning using agent influence. In *2021 IEEE Conference on Games (CoG)*, pages 01–08. IEEE, 2021.

[14] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992.

[15] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[17] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[18] Marcelo L Koga, Valdinei Freire, and Anna HR Costa. Stochastic abstract policies: Generalizing knowledge to improve reinforcement learning. *IEEE Transactions on Cybernetics*, 45(1):77–88, 2014.

[19] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[20] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[21] Tianqi Chen, Ian Goodfellow, and Jonathon Shlens. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*, 2015.

[22] Dongsheng Xu, Peng Qiao, and Yong Dou. Aggregation transfer learning for multi-agent reinforcement learning. In *2021 2nd International Conference on Big Data & Artificial Intelligence & Software Engineering (ICBASE)*, pages 547–551. IEEE, 2021.

[23] Muhammad Junaid Khan, Shah Hassan, and Gita Sukthankar. Leveraging transformers for starcraft macromanagement prediction. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1229–1234. IEEE, 2021.

[24] Ben Tan, Yangqiu Song, Erheng Zhong, and Qiang Yang. Transitive transfer learning. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1155–1164, 2015.

[25] Ruo-Ze Liu, Haifeng Guo, Xiaozhong Ji, Yang Yu, Zhen-Jia Pang, Zitai Xiao, Yuzhou Wu, and Tong Lu. Efficient reinforcement learning for starcraft by abstract forward models and transfer learning. *IEEE Transactions on Games*, 14(2):294–307, 2021.

[26] Kun Shao, Yuanheng Zhu, and Dongbin Zhao. Starcraft micromanagement with reinforcement learning and curriculum transfer learning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(1):73–84, 2018.

[27] Christopher D Rosin and Richard K Belew. New methods for competitive coevolution. *Evolutionary computation*, 5(1):1–29, 1997.

[28] Shimon Whiteson. Evolutionary computation for reinforcement learning. *Reinforcement Learning: State-of-the-art*, pages 325–355, 2012.

[29] Bi Li, Tu-Sheng Lin, Liang Liao, and Ce Fan. Genetic algorithm based on multipopulation competitive coevolution. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 225–228. IEEE, 2008.

[30] Jian-guo Liu. Competitive coevolutionary genetic algorithms for multi-objective optimization problems. In *2009 International Conference on Artificial Intelligence and Computational Intelligence*, volume 3, pages 594–597. IEEE, 2009.

[31] David Cotton, Jason Traish, and Zenon Chaczko. Coevolutionary deep reinforcement learning. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2600–2607. IEEE, 2020.

[32] Megan M Olsen and Rachel Fraczkowski. Co-evolution in predator prey through reinforcement learning. *Journal of computational science*, 9:118–124, 2015.

[33] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. PMLR, 2017.

[34] Marcin Szubert, Wojciech Jaśkowski, Paweł Liskowski, and Krzysztof Krawiec. The role of behavioral diversity and difficulty of opponents in coevolving game-playing agents. In *Applications of Evolutionary Computation: 18th European Conference, EvoApplications 2015, Copenhagen, Denmark, April 8-10, 2015, Proceedings 18*, pages 394–405. Springer, 2015.

[35] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[36] Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*, 2017.

[37] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

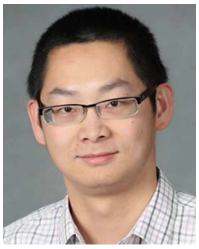
[38] Ayesha Siddika Nipu, Siming Liu, and Anthony Harris. Maidcrl: Semi-centralized multi-agent influence dense-cnn reinforcement learning. In *2022 IEEE Conference on Games (CoG)*, pages 512–515, 2022.

[39] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

[40] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu.

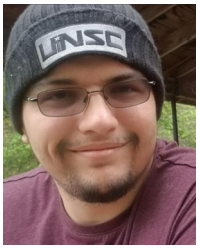
Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.
[41] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.
[42] Abhishek Gupta, Yew-Soon Ong, and Liang Feng. Insights on transfer optimization: Because experience is the best teacher. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1):51–64, 2017.

AYESHA SIDDIQUA received her B.S. degree in Computer Science from Bangladesh University of Engineering and Technology in 2019. She is currently a master's student in the Department of Computer Science at Missouri State University. Her research interests include artificial intelligence, machine learning, reinforcement learning, and multi-agent systems.



SIMING LIU (Member, IEEE) received the Ph.D. degree in Computer Science and Engineering from the University of Nevada, Reno, in 2015. He is currently an Assistant Professor of Computer Science at Missouri State University (MSU). He has been serving as the director of the Security and Artificial Intelligence Laboratory at MSU since 2019. His research interests include computational intelligence, reinforcement learning, and evolutionary computation, with a focus on applications in computer game AI, simulations, and multi-agent systems.

AYESHA SIDDIKA NIPU earned her Master's degree in Computer Science from Missouri State University in 2023. She is currently serving as a Lecturer in the Department of Computer Science and Software Engineering at the University of Wisconsin-Platteville. Her research interests include, but are not limited to, Machine Learning, Artificial Intelligence, Natural Language Processing, and Generative AI.



ANTHONY HARRIS received the masters degree in the Department of Computer Science from Missouri State University, in 2021. He joined O'Reilly Automotive, Inc. as a Software Developer in 2020 and currently holds the position of Data Scientist. His research interests include computational intelligence, reinforcement learning, and evolutionary computation, with a focus on applications in computer game AI, simulations, human behavior modeling, and multi-agent systems.

YAN LIU received the Ph.D. degree in Computer Science and Technology from Hunan University, China in 2010. He is currently an Associate Professor with the College of Computer Science and Electronic Engineering, Hunan University. His research interests include embedded systems, internet of things, cybersecurity, artificial intelligence, and parallel and distributed systems, with a focus on applications in self-driving cars and smart environments.