Learning Diverse Policies in MOBA Games via Macro-Goals

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently, researchers have made successful progress on MOBA games, such as Dota 2 and *Honor of Kings*. Even though they have achieved or even exceeded human-level performance, the methods with deep reinforcement learning still suffer from the lack of policy diversity. In this paper, we propose a novel framework that can learn diverse policies in MOBA games. We abstract strategies as macro-goals from human experience and then train a Meta-Controller to predict these macro-goals. To enhance policy diversity, we sample strategies from the Meta-Controller and guide the training of policies towards macro-goals. The experimental results demonstrate that our methods outperform the state-of-the-art systems of over 102 heroes in *Honor of Kings*. Our agents can also exploit diverse policies in different matches and lineups.

1 Introduction

As a stepping stone of Artificial General Intelligence (AGI), Game AI has achieved significant progress in the past decade, including board games Silver *et al.* [2016], poker games Brown *et al.* [2019]; Moravčík *et al.* [2017], real-time strategy (RTS) games Vinyals *et al.* [2019], etc. Especially, Multi-player Online Battle Arena (MOBA) game has been regarded as a challenging problem in Game AI because of its complicated mechanisms such as imperfect information, large state-action space, and long-term sequential decision making. MOBA games have attracted massive attention from research communities. For example, OpenAI develops the AI system OpenAI Five which defeats the Dota 2 world champion, i.e., Team OG, within a limited size of hero pool Berner *et al.* [2019]. It trains with deep reinforcement learning (DRL) and self-play. Ye *et al.* [2020a] build AI agents on a popular MOBA game named *Honor of Kings*, which also defeat top esports players. Compared with OpenAI Five, their method enables playing full MOBA games, which extends the hero pool size from 17 to 40 heroes, adds ban/pick capabilities, and addresses the scalability issue.

Even though existing AI systems have achieved or even exceeded human-level performance in MOBA games, their agents still suffer from the lack of policy diversity. Agents during training are controlled by the same hand-crafted reward signal and then present similar strategies even in different matches. However, strategies played by humans usually vary according to situations in the game. One of the disadvantages is that we can counter the single learned policy after several attempts effortlessly. For example, the same team wins OpenAI Five nine times with the same strategy in public tests[1]. The primary counter-strategy is to avoid fights as much as possible and instead push out side lanes to pressure the agents into defending[2].

Another disadvantage is that agents with a single policy cannot exploit the strength of heroes and lineups. To interest players, MOBA game designers usually make different skill mechanisms for

[1] https://openai.com/projects/five/

[2] https://www.reddit.com/r/MachineLearning/comments/bfq8v9/

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

different heroes, which result in heroes play different roles in the game. Therefore, different roles in lineups have different strategies to maximize a team's utility. For example, the supporter should keep their allies alive and give them opportunities to earn more gold and experience. The assassin can efficiently jungle neutrals rather than farm lanes. Moreover, different combinations of heroes construct lineups with different characteristics. Some lineups have advantages in pushing with a quick pace, while some lineups are good at the development of carry heroes. The agent should exploit the picked lineup's strength instead of playing with a single policy in the game.

Although existing methods can learn diverse policies in other games, it remains to be a grand challenge in MOBA games. AlphaStar Vinyals *et al.* [2019] uses a framework based on the league to improve the diversity of policies in StarCraft II. They construct the league with main agents, main exploiters, and league exploiters to keep diverse policies during training. They also define a latent variable according to building orders in human data as constraints of self-play. Unfortunately, their methods cannot be applied to MOBA games directly. First, their methods consume massive computation resources to construct the league. They use 12 copies of agents in which there exist 128 TPU cores. Second, the scenarios between MOBA and RTS games are different. They leverage the nature of StarCraft II and use building orders to represent each strategy, which is unavailable in MOBA games.

In this paper, we propose a novel learning paradigm named **M**acro-**G**oals **G**uided (MGG) learning to train diverse policies in MOBA games. Motivated by top players, we define related game information as macro-goals to depict strategic decisions, which constitute macro strategies in MOBA games. To explore macro strategies efficiently, we extract macro-goals from human experience and train a supervised model to predict macro-goals according to the given lineups and game information. We sample macro goals from the model and incorporate them into the original actor-critic framework. We design an intrinsic reward that guides the trained policy to achieve the generated macro-goals. In this paper, we use the *Honor of Kings*[3], a famous mobile 5v5 MOBA game, as our testbed. The experimental results demonstrate that our AI agent can present diverse policies according to the given lineups and improve the Elo score by modeling policy diversities. Our contribution can be concluded as follows:

- To the best of our knowledge, we are the first to investigate the problem of policy diversity in MOBA games. We propose a novel framework that can learn diverse policies to adapt to different matches and lineups.
- Based on macro-goals, our methods can explore diverse macro strategies efficiently and meanwhile preserve high level micro-operations. Unlike league-based methods, our MGG framework uses the same computation resources as the original actor-critic framework.
- We conduct performance testing and diversity analysis on the full hero pool, i.e., 102 heroes. Experimental results show that our agents can outperform existing state-of-the-art systems and demonstrate diverse policies.

2 Related Works

2.1 Artificial Intelligence in MOBA Games

The progress of Game AI in MOBA mainly focuses on Dota 2 and *Honor of Kings*. OpenAI Five has successfully defeated top esports players on playing 1v1 and 5v5 games in Dota 2 within limited heroes Berner *et al.* [2019]. Ye *et al.* [2020b,c,a] has proposed a series of works that exceed human-level performance and expand hero pools in *Honor of Kings*. Besides, TencentHMS Wu [2019] exploits a hierarchical policy structure to control macro strategy decisions and micro-level executions, respectively. All the mentioned works except the SL model adopt a single strategy regardless of the heroes and lineups, which cannot take full advantage of heroes and lineups' characteristics.

2.2 Diverse Policies in Other Games

The league-based methods Vinyals *et al.* [2019]; Balduzzi *et al.* [2019]; Jaderberg *et al.* [2019, 2017] train multiple agents simultaneously to explore strategy space and require massive computation resources. However, it is impractical to use league-based methods in MOBA games. Because the

[3]https://en.wikipedia.org/wiki/Honor_of_Kings

space of strategies is complex in that there are $213,610,453,056(C_{40}^{10} \times C_{10}^{5})$ lineups for 40 heroes in MOBA games and each lineup has a different strategy. Moreover, there is no obvious signal, e.g., building order, to represent macro strategies in MOBA games.

The aforementioned Game AIs design methods suffer from the following limitations: 1) the need for data on human behaviors; 2) heavy dependence of designer expert knowledge and substantial labor costs in searching a desirable behavior. Shen *et al.* [2020] can automatically generate diverse styles with almost no domain knowledge. However, this method is not suitable for complex games that last up to an hour long and consist of thousands of actions, such as MOBA games and RTS games.

2.3 Imitation Learning

In order to reduce the cost of strategy exploration in enormous lineups, we imitate macro-strategies from human experience. Actually some works have trained reward functions from human preferences Nair *et al.* [2018]; Ibarz *et al.* [2018]; Christiano *et al.* [2017] or learnt policy directly from limited human data Ho and Ermon [2016]. Different from the above works, we abstract strategy as macro-goals in MOBA games. Our method not only imitates macro-strategies, but also leaves the agent to explore the optimal micro-operations (e.g., move, attack) by itself through external rewards.

2.4 Goal-based Learning

Goal-based methods Schaul *et al.* [2015]; Andrychowicz *et al.* [2017] are originally proposed to solve the problem of training a single goal-conditioned policy to accomplish multiple tasks. Some methods refer to the idea of goal-based RL and divide a task into a series of sub-tasks (i.e., goal) to accelerate the exploration and training of policy, such as automatic goal generation Florensa *et al.* [2018] and hierarchical RLKulkarni *et al.* [2016]; Nachum *et al.* [2018], etc. However, in complex scenarios such as MOBA games or RTS games, it is difficult to explore the meaningful goals, whether using GAN or training a high-level policy by RL methods. Therefore, we utilize the top players' demonstrations to pre-train a macro-level policy to generate reasonable and diverse macro-goals.

3 Macro-Goals Guided Reinforcement Learning

Our problem can be expressed as the form of Markov Decision Process (MDP), which consists of a 6-tuple (S, G, A, R, T, γ), where S is the set of states, A is the action space, $T(S, A, S)$ is the transition function, R is the reward function, and $\gamma \in [0, 1]$ is a discount factor. As described by Silva and Chaimowicz [2017], in a typical MOBA game, the agents need to perform actions $a \in A$ (e.g. moving, attacking) according to the observations $s \in S$ (e.g. unit states, game stats) to achieve the final objective (e.g. destroying the crystal).

In particular, we introduce G which represents the macro strategic states space in the game, where the current macro state is $c = f(s) \in G$, and the macro goal is $g \in G$, where f is the macro state extraction function $S \rightarrow G$. Since the general reward function $R(S, A)$ is designed for the common strategy, its macro state set $\{c|c \in G_{RL}\}$ cannot fully cover G, i.e. $G_{RL} \subset G$. However, human strategies are diverse, so the human macro state set G_{human} is considered to be close to G, i.e. $G_{human} = G_1 \cup \ldots G_i \cdots \cup G_N \approx G$, where N is the number of human strategies. We define that strategy i can be expressed as a set of macro state $\{c|c \in G_i\}$, where $G_i \subset G$. Therefore, we model the policy as $\pi(a|s, g)$, and our objective is to use $g \in G_{human} \approx G$ as the macro goal of the policy and use intrinsic reward $R(f(S), G, A)$ to guide the macro state of the policy to be close to the complete G, that is, to maximize the marco state entropy $-\sum_{c \in G} P(c) log P(c)$.

In this paper, we propose a **M**acro-**G**oals **G**uided (MGG) framework to learn diverse policies in MOBA games. Top players' matches are observed to have high quality of diverse macro-strategies but noisy micro-operations. Thus, we imitate strategies from top players' matches to reduce the cost of exploration in our training. Completely imitating human data including micro-operations will worsen the performance of our policy. Therefore, our motivation is to define macro-goals representing human strategies and then learn diverse strategies from these macro-goals only. The MGG method consists of three stages, as shown in Figure 1.

Stage 1: Game state s_t^D and its corresponding goal g^D are extracted from top players' demonstrations.

Stage 2: A network ϕ is designed to predict macro-goals, i.e., Meta-Controller. The extracted state and macro-goal pairs $\langle s_t^D, g^D \rangle$ are used as the training data and then the ϕ is trained via supervised learning. During the training, the auxiliary tasks T_{aux} are introduced to improve predicting effects. Given a state s, the ϕ should predict a macro-goal g to depict the current strategy.

Stage 3: The Dual-clip PPO method Ye *et al.* [2020a] is adopted to train the policy and value network conditioned on the goal g. Given state s_t, its corresponding goal g could be obtained via pre-trained Meta-Controller. The input of the policy and value network θ contains both macro-goal g and s_t. A new reward R_t^{goal} is defined to denote whether the trained agent achieves g. Through the multi-head value mechanism Ye *et al.* [2020a], the value calculated by R_t^{goal} and the original value are combined.

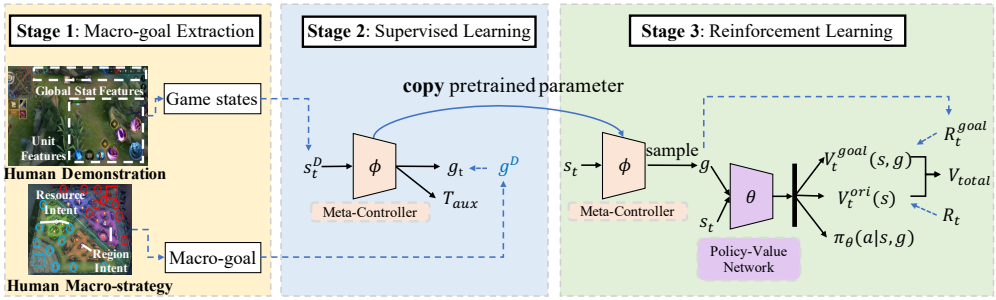


Figure 1: The framework of our MGG method.

To better illustrate the MGG method, we use *Honor of Kings* as a case study to describe some common terms in MOBA games. The MGG method also works with other MOBA games, as the mechanisms of MOBA games (e.g. Dota 2 and League of Legends) are similar.

3.1 Generation of Macro-Goals

3.1.1 Macro-goal Definition

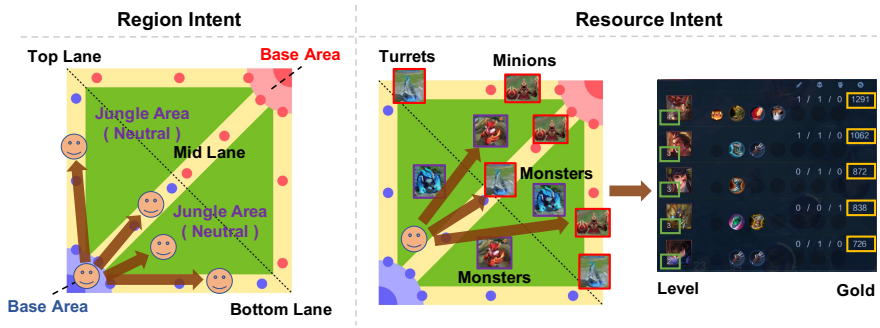


Figure 2: Macro-goal designed for abstracting player's macro-strategy. The red and blue on the map indicate two teams respectively, and the map can be divided into 3 *lanes* (the top, middle and bottom lane), 4 *jungle areas*, and 2 *base areas*. Circles represent *turrets*. The brown arrows on the left indicate region intent ("Where to go"). The *turrets*, *monsters* and *minions* on the right represent unit resource. The brown arrows on the right indicate resource intent, which means that the agent will kill units (" what to do ") in order to level up or gain gold.

Our motivation for designing MOBA AI macro-goals is inspired from the way human players make strategic decisions. Human players usually make strategic decisions according to "Where to go" and "What to do". Thus, we define two types of macro-goals for all MOBA games, i.e., region intent ("Where to go") and resource intent ("What to do"), as shown in Figure 2. The region intent refers to the next region where the players move to, as shown in the left part of Figure 2. The resource intent refers to the quantity of game resources that players hope to obtain, including turrets, monsters, gold assignment, and other common resources in MOBA games, as shown in the right part of Figure 2.

3.1.2 Macro-Goal Label Extraction

Macro-goals can be extracted from top players' matches. Given a state s_t in the trajectory, we define a multi-dimensional goal and extract its region intents and resource intents in the future frame $f(s_{t+C})$ as labels of macro-goals g_t. Each dimension denotes an aspect of the goals in the current state. The multi-dimensional goal directs policy to learn to achieve multiple intents at the same time. Figure 3

shows a detailed example of macro-goal label extraction. The trajectory consists of frames in matches of top players. Let T_s denote the current frame. Therefore, we can sample the frame with goals from the interval $[T_s + C - \epsilon, T_s + C + \epsilon]$, where C is the delta of frame numbers between the state and its goals. ϵ is a hyper-parameter representing the range of noise to avoid overfitting. Theoretically, extracting labels in this way, we expect our Meta-Controller to learn the mapping from the current state s_t to its future macro-goal g_t.

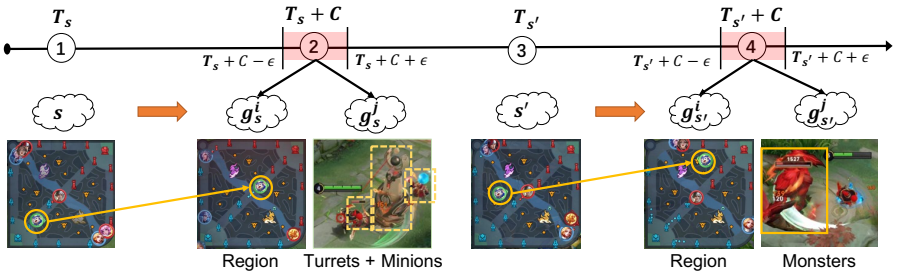


Figure 3: An example of macro-goal label extraction. The macro state $f(s_{t+C})$ at $t + C$ is taken as the macro goal of the state s_t at t. The macro-goals at T_s is to move into the opposite middle lane and gain gold resources from turrets and minions, while at $T_{s'}$ is to move to the upper jungle area and grab the monsters to level up.

3.1.3 Meta-Controller Training

After extracting macro-goals, the prediction model Meta-Controller is trained in a supervised way. The network structure of Meta-Controller is shown in Figure 4. To predict macro-goals, we input current game state features including observable unit's attributes and in-game statistics information, e.g. health point(hp), location, gold, level, etc. Note that the game state features include information of *heroes* and *lineups*, and the lineup is a combination of five heroes. In this way, the Meta-Controller can predict macro-goals of different strategies according to the lineup features.

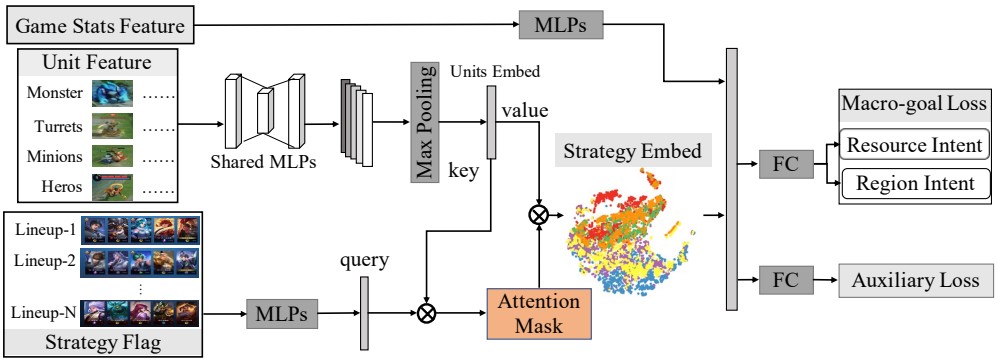


Figure 4: Network Architecture of the Meta-Controller. First, the features of each unit are extracted by some shared MLPs into units embedding. Then, the lineup information is encoded by MLPs and used as a query vector to calculate the attention between the strategy and units embedding. Finally, the strategy embedding and game stats embedding are used to predict macro goals $\phi(s_t)^g$ and strategy flag $\phi(s_t)^{aux}$.

Generally, strategies in MOBA games are closely related to lineups. Enabling the predicted macro-goals to denote strategies, we regard the auto-encoding task of lineups as an auxiliary task T_{aux}, which is essential to learn latent representations of strategies. Then, the attention mechanism is adopted. The latent representations of strategies are regarded as queries and the hidden encoding of game states are regarded as keys and values. In addition, to deal with the imbalance of macro-goal labels and improve the prediction accuracy, we use the focal loss denoted as L_{FL} Lin *et al.* [2017]. The loss function is:

$$L^{SL}(\phi) = L_{FL}(\phi(s)^g, y^g) + \lambda * L_{CE}(\phi(s_t)^{aux}, y^{aux}) \tag{1}$$

where L_{FL} is focal loss of predicting labels and L_{CE} is cross-entropy loss of the auxiliary task. λ is a balancing parameter. We use Adam Kingma and Ba [2014] to solve the optimization problem.

3.2 Macro-goals Guided Reinforcement Learning

We integrate the Meta-Controller into the original actor-critic framework as shown in Figure 5. Given the state s_t at step t, the pre-trained Meta-Controller generates a macro-goal g_t. To guide the trained

policy towards g_t, inspired by goal-based RL Andrychowicz *et al.* [2017]; Schaul *et al.* [2015], the policy and value function make decisions according to not only s_t but also g_t. For every N frames, the Meta-Controller generates a new goal to follow the strategy. Therefore, we estimate parameters of policy $\pi_\theta(a_t|s_t, g_t)$ and value functions $V_\theta(s_t, g_t)$ during reinforcement learning training. Note that in order to increase policy diversity, we sample macro-goals from the softmax distribution of Meta-Controller instead of selecting according to argmax function. Then, like human players, we have variant strategies even the lineups are the same.

We design a new reward R^{goal} to guide the policy towards the macro-goals as follows:

$$R^{goal}(s_t, a_t, s_{t+1}, g_{t'}) = \left|f(s_t) - g_{t'}\right| - \left|f(s_{t+1}) - g_{t'}\right| \tag{2}$$

where $g_{t'}$ is the generated macro-goal $\phi(s_t)^g$. f is the mapping from the current state to the macro state, which is the same with the label extraction process. We define the intrinsic rewards R^{goal} as the delta of the distance between the current state and the macro-goal. If the state s_{t+1} is closer to $g_{t'}$ than s_t, the agent will receive a positive reward.

The intrinsic reward R^{goal} is combined with the environment reward, so that the agent is trained to maximize the accumulated environment rewards (RL objective) as well as performing macro-goal distribution matching (IL objective). In this way, when guiding agents to execute human macro-strategies, agents need to explore their own way to achieve the goals, instead of being guided at a micro-operation level like imitation learning methods, which may weaken their ability.

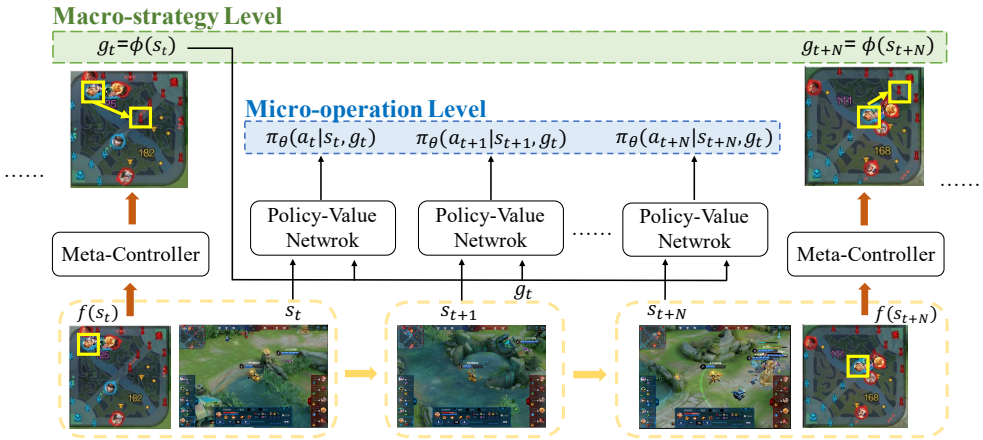


Figure 5: An example of a trajectory with MGG.

Policy updates. In order to avoid training instability in a large-scale distributed environment, we use the Dual-clip PPO method Ye *et al.* [2020a]. Different from the original algorithm, the macro-goal g in introduced into the policy $\pi_\theta(a_t|s_t, g_t)$ and advantage estimation $\hat{A}_t(a_t, s_t, g_t)$ in Dual-clip PPO. When $\pi_\theta(a_t|s_t, g_t) \gg \pi_{\theta_{old}}(a_t|s_t, g_t)$ and $\hat{A}_t < 0$, the radio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t,g_t)}{\pi_{\theta_{old}}(a_t|s_t,g_t)}$ is a huge number, which will introduce a big and unbounded variance because $r_t(\theta)\hat{A}_t \ll 0$. Dual-clip PPO introduces another clipping parameter c that indicates the lower bound when $\hat{A}_t < 0$. The new objective is the following:

$$L^{policy}(\theta) = \hat{\mathbb{E}}_t[\max(c\hat{A}_t, \min(clip(r_t(\theta), 1-\tau, 1+\tau)\hat{A}_t, r_t(\theta)\hat{A}_t))]$$

where τ is the original clip in PPO.

Value updates. We use multi-head value mechanism Ye *et al.* [2020a] to incorporate the value of macro-goal. Specifically, the value of achieving macro-goals are regarded as one head. Therefore, the value loss is defined as follows:

$$L^{value}(\theta) = \hat{\mathbb{E}}_t[\sum_{head_k}(R_t^k - \hat{V}_t^k)], V_{total} = \sum_{head_k} w_k V_t^k(s_t, g_t)$$

where w_k is the weight of the kth head and $V_t^k(s_t, g_t)$ is the kth value.

4 Experiments

In this section, we first compare the state-of-the-art methods of supervised learning (SL) Ye *et al.* [2020b] and reinforcement learning (RL-baseline) Ye *et al.* [2020a] in *Honor of Kings* with the MGG

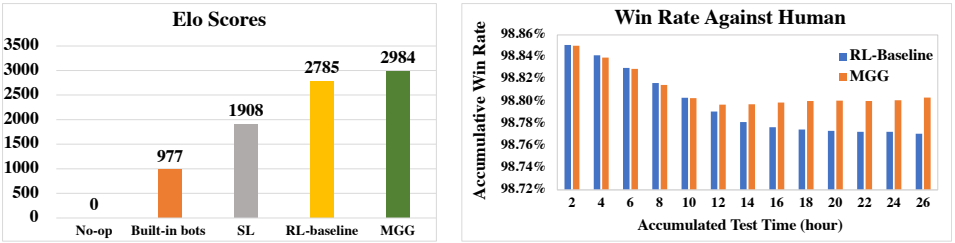
method, from the perspective of capability and diversity. Then, in order to evaluate the ability gains from increasing strategic diversity, several special lineup systems are selected to compare the effects of MGG, RL-baseline and GAIL Ho and Ermon [2016] . Since GAIL needs human data which will greatly affect the sample efficiency, making it impossible to be trained on 102 heroes, we only make comparisons with it in the case study (approximately 20 heroes for each case).

4.1 Experiment Settings

The dataset we use to train the Meta-Controller contains replays of the top 1% human players in the *Honor of Kings*. We extract 4060-dimensional state and 64-dimensional macro-goal pairs $\langle s_t^D, g^D \rangle$ from the raw data. Empirically, we set C to 30 seconds and ϵ to 3 seconds. For all AI heroes, only one Meta-Controller model needs to be trained. 8 NVIDIA P40 GPUs is used to train it for about 26 hours, and the batch size of each GPU is set to 512. We set $\alpha = 0.75, \gamma = 2$ for focal loss L_{FL} Lin *et al.* [2017], and $\lambda = 1$ for the weight of auxiliary task. Adam is adopted with the initial learning rate of 0.0001. Finally, the macro-goal is predicted via a convergence model of 1 million steps.

To make a fair comparison, MGG and other RL baselines run over a physical computing cluster with 60,000 CPUs and 830 NVIDIA V100 GPUs. The batch size of each GPU is set to 4096. After 420 hours of training, the models of all the algorithms converge. We evaluate all the methods on 102 heroes[4] in Section 4.2 and 4.3, and on about 20 heroes in Section 4.4. For online evaluation, AI's response time is set to 193ms, including observation delay (133 ms) and response delay (about 60 ms). Response delay consists of features, models, result processing, and transmission delays.

4.2 AI Performance



(a) (b)

Figure 6: The performance comparison between MGG and the existing AI sytems in the *Honor of Kings*. (a) Comparison of ELO scores. (b) Win rate comparison in large-scale human-machine tests.

MGG is compared with built-in bots, supervised learning agents using human game data Ye *et al.* [2020b], and the original reinforcement learning agents(RL-baseline) Ye *et al.* [2020a]. Note that RL-baseline is the state-of-the-art method in the *Honor of Kings*. We run 400 matches for each comparison and calculate the Elo score Coulom [2008]. From Figure 6(a), it is concluded that the MGG method is superior to other baseline methods in real games, for it benefits from modeling policy diversity and being able to take advantage of the lineups' characteristics.

Moreover, a large-scale human-machine test(about 100,000 tests in 26 hours) was conducted on April 3, 2021 to demonstrate the impact of policy diversity on performance. In these test, human players are randomly matched against the RL-baseline and MGG model, respectively. Human players are told that the opponent is the AI but do not know what the specific model is. As shown in Figure 6(b), the human players gradually find strategies to defeat the AIs as the test time increased, resulting in a decrease in the AI's win rate. However, after 11 hours, as the number of strategies tried by humans increases, the win rate of MGG tends to be stable, while that of the RL-baseline continues to decline. Thus, this test effectively proves that diverse policies are superior to single-strategy policies because the latter are more likely to be countered.

4.3 Diversity in Different Lineups and Matches

4.3.1 Diversity in Different Lineups

Figure 7 shows the visualization result of macro state embeddings of MGG, RL-baseline and human, and points are colored according to different methods. MGG plays more diverse policies than RL-baseline. The macro state space of MGG (blue points) covers both the RL-baseline (red points) and human (yellow points), which indicates MGG can play both the human strategies and RL-baseline

[4] see Supplementary for more details.

strategies. Besides, it is observed that the point clusters of MGG and human strategy are obviously separated from each other. Conversely, the points of the RL-baseline strategy are completely mixed.

Then, we evaluate the Davies-Bouldin Index Davies and Bouldin [1979] of marco states embeddings in different lineups as shown in Table 1. The MGG has a lower Davies-Bouldin index, indicating there is more variation in MGG's macro-strategies among different lineups.

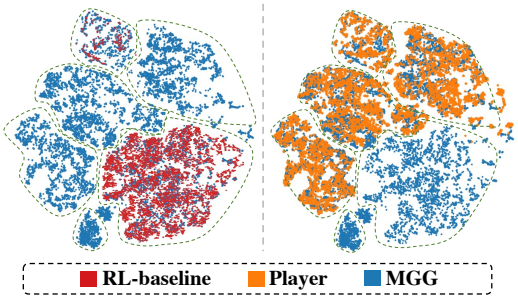


Figure 7: **Macro states t-SNE Embedding of different algorithms in the same space**. **Left:** comparison of MGG (blue points) and RL-baseline (red points). **Right:** comparison of MGG and human player (yellow points). The strategy space of MGG covers both.

Table 1: Davies-Bouldin Index of MGG and RL-baseline in different lineups. The values closer to zero indicate a better partition, meaning that the strategies (macro state space) varies more between the different lineups.

	Davies-Bouldin Index
RL-baseline	10.72
MGG	1.76
human	1.45

4.3.2 Diversity in Different Matches

We further evaluate the diversity for the same lineup in different matches. The marco state entropy metric is defined as $H = -\sum_{c \in G} P(c) log P(c)$, where $P(c)$ is the probability mass function of marco state and $c = f(s)$. In this experiment, we let agents play against the same built-in bot with the same lineup and the match starts from the same state. Each comparison runs 1000 matches.

Table 2: Comparing state-entropy of each AI system in the different matches when they pick the same lineup.

	Built-in bots	RL-baseline	MGG
state entropy	0.0	0.014	**0.408**

Table 2 shows the comparison results between MGG and RL-baseline, which indicates that the MGG agent can play diverse polices even in the same lineup due to sampling the macro-goal from the Meta-Controller in real time.

4.4 Case Study

In the *Honor of Kings*, the most common opening-strategy for human players is three-lane-strategy, in which the marksman, mage, and warrior (or tank) obtain resources from three lanes respectively, while the assassin grabs its own neutral jungle resources, and the supporter roams to assist them without gaining resources. Since the general reward function Ye *et al.* [2020a] is designed based on this strategy, agents will execute similar strategies in different lineups and cannot exploit the most suitable strategy for many special lineups. To evaluate the effectiveness of MGG on this issue, we select several lineup systems[5] requiring unconventional strategies to compare the diversity measures and capabilities of MGG and other approaches.

In addition, we compare MGG with GAIL Ho and Ermon [2016], which is recently proved to be effective in behavior imitating tasks. GAIL uses a discriminator to distinguish whether a state-action pair is from the human player or the learned policy, and optimizes the policy to confuse the discriminator in the meanwhile. However, adversarial training will greatly affect the training efficiency of the policy, so we only train GAIL under certain special lineup systems, instead of the full hero pool.

4.4.1 Case of Marksman-Core System

In the *Marksman-Core System*, mage is replaced with supporter so that the marksman can obtain the resources from two lanes to gain an advantage over the enemy in gold and equipment rapidly. Therefore, we need to evaluate the degree of completion of the team resource allocation strategy from two aspects: (1) the gold share of marksman in the team($\frac{gold_{marksman}}{\sum(gold_{team_index})}$); (2) the gold gap

[5] see Supplementary for more cases of lineup systems and results of their human-machine test.

of marksman and enemy team($gold_{marksman} - \frac{1}{5}\sum(gold_{enemy_index})$). As shown in Figure 8, the distribution of MGG is the closest to that of human players in both aspects, indicating that applying MGG is more beneficial to achieve the resource intent strategies (e.g. resource allocation), which is difficult for the original reward function.

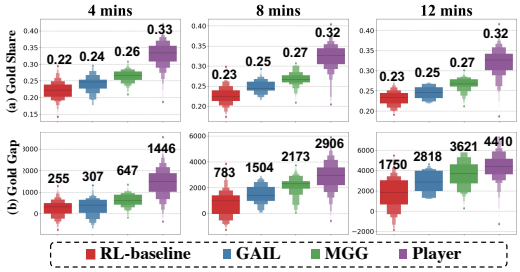


Figure 8: Statistical indicator of resource allocation in *Marksman-Core System*.

Table 3: Neutral monster resource metrics of *Resource-Grab System*. In the early stage, the near side resources are at most 2 and the far side resources are also capped at 2.

	Near Side Resources	Far Side Resources	Total Resources
human	1.317	0.561	1.878
RL-baseline	1	0	1
GAIL	1.141	0.056	1.197
MGG	**1.741**	**0.658**	**2.399**

4.4.2 Case of Resource-Grab System

In the *Resource-Grab System*, a assassin strong in the early stage(first 2 minutes) is usually chosen to grab the neutral monster resources closer to the opponents. This system belongs to a long-term regional intent strategy. Therefore, the success of the strategy can be denoted by the number of neutral resources grabbed in the early stage. As shown in Table 3, by using conventional strategy, RL-baseline just maximizes the short-term gains by only grabbing the resources close to its own side. For GAIL, it is also difficult to execute the long-term strategy because GAIL only guide agents at the micro-operation level. In contrast, MGG can effectively teach the agents to execute the long-term strategy and outperforms the human by 0.52 monster resources in total.

4.4.3 Capability Analysis

To further investigate whether increased diversity can improve the capability of special lineup systems, we conduct an experiment by letting these methods play against in pair, 100 times for each pair. All methods use models whose capabilities have converged. Figure 9 compares the Elo scores of built-in bots, supervised learning agents, the RL-baseline agents, GAIL agents, and MGG agents. It indicates different strategies significantly affects the upper limit of the capability for the same lineup. As strategy diversity improves, MGG increases 312 and 560 Elo scores over baseline in the above two lineup systems, respectively. Overall, the method we proposed contributes to training MOBA AI with more diverse strategies and enhancing capabilities of special lineup systems. The video of the above case can be found in https://sites.google.com/view/mgg-demo.

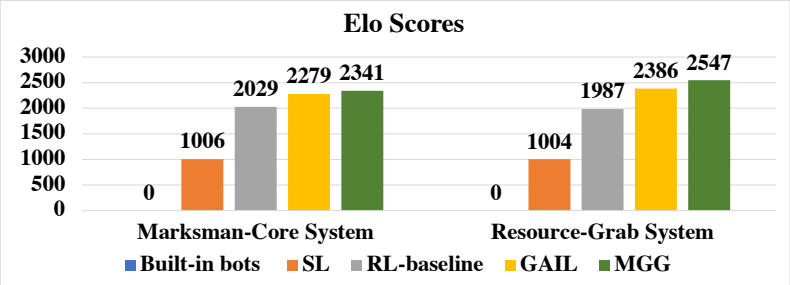


Figure 9: Comparison of Elo scores between MGG and other methods in two special lineup systems.

5 Conclusions

In this paper, a MOBA AI framework is proposed, which can learn diverse policies during training. Macro-goals are first defined to depict diverse strategies in MOBA games. Then, by sampling macro-goals from Softmax distribution of Meta-Controller, policy diversity is increased. Moreover, to guide the policy towards a specific strategy, we design an intrinsic reward to indicate whether the agents achieve macro-goals during training. As a result, the trained agents can play diverse policies in different matches and exploit lineups' characteristics to improve capabilities. For future work, after learning diverse policies, we will investigate non-transitivity in strategies of MOBA games using game theory. Also, we expect to find a solution which is closer to Nash equilibrium in MOBA games.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in neural information processing systems*, pages 5048–5058, 2017.

David Balduzzi, Marta Garnelo, Yoram Bachrach, Wojciech M Czarnecki, Julien Perolat, Max Jaderberg, and Thore Graepel. Open-ended learning in symmetric zero-sum games. *arXiv preprint arXiv:1901.08106*, 2019.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International conference on machine learning*, pages 793–802, 2019.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in neural information processing systems*, pages 4299–4307, 2017.

Rémi Coulom. Whole-history rating: A bayesian rating system for players of time-varying strength. In *International Conference on Computers and Games*, pages 113–124, 2008.

D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (2):224–227, 1979.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pages 1515–1528. PMLR, 2018.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *arXiv preprint arXiv:1606.03476*, 2016.

Borja Ibarz, Jan Leike, Tobias Pohlen, Geoffrey Irving, Shane Legg, and Dario Amodei. Reward learning from human preferences and demonstrations in atari. In *Advances in neural information processing systems*, pages 8011–8023, 2018.

Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science*, 364(6443):859–865, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Tejas D Kulkarni, Karthik R Narasimhan, Ardavan Saeedi, and Joshua B Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *arXiv preprint arXiv:1604.06057*, 2016.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

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.

Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *arXiv preprint arXiv:1805.08296*, 2018.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6292–6299, 2018.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320, 2015.

Ruimin Shen, Yan Zheng, Jianye Hao, Zhaopeng Meng, Yingfeng Chen, Changjie Fan, and Yang Liu. Generating behavior-diverse game ais with evolutionary multi-objective deep reinforcement learning. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (1JCA1)*, pages 3371–3377, 2020.

Victor do Nascimento Silva and Luiz Chaimowicz. Moba: a new arena for game ai. *arXiv preprint arXiv:1705.10443*, 2017.

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.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

Bin Wu. Hierarchical macro strategy model for moba game ai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1206–1213, 2019.

Deheng Ye, Guibin Chen, Wen Zhang, Sheng Chen, Bo Yuan, Bo Liu, Jia Chen, Zhao Liu, Fuhao Qiu, Hongsheng Yu, et al. Towards playing full moba games with deep reinforcement learning. *arXiv preprint arXiv:2011.12692*, 2020.

Deheng Ye, Guibin Chen, Peilin Zhao, Fuhao Qiu, Bo Yuan, Wen Zhang, Sheng Chen, Mingfei Sun, Xiaoqian Li, Siqin Li, et al. Supervised learning achieves human-level performance in moba games: A case study of honor of kings. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, et al. Mastering complex control in moba games with deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6672–6679, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See the assumptions in Section 3.
   (c) Did you discuss any potential negative societal impacts of your work? [No] For the research community, we propose a general framework for learning diverse policies in MOBA games. We herewith expect this work to provide inspirations to other complex real-world problem.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code and the data are proprietary.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.1.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4.1.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [No] The assets are private and we have permission from the creators.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] See Section 4.2.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The game information does not contain any identity information.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [Yes] We declare the test instructions in the game.

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [Yes]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [Yes] We give out in-game rewards for testing.