Complex Instruction Following with Diverse Style Policies in Football Games

Chenglu Sun[1*], **Shuo Shen**[1*], **Haonan Hu**[1], **Wei Zhou**[2], **Chen Chen**[3†]

[1]Sports Products Department, Interactive Entertainment Group, Tencent
[2]School of Future Technology, Nanjing University of Information Science and Technology
[3]Human Phenome Institute, Fudan University
{clivesun, svenshen, rubenhnhu}@tencent.com, wei.zhou@nuist.edu.cn, chenchen_fd@fudan.edu.cn

Abstract

Despite advancements in language-controlled reinforcement learning (LC-RL) for basic domains and straightforward commands (e.g., object manipulation and navigation), effectively extending LC-RL to comprehend and execute high-level or abstract instructions in complex, multi-agent environments, such as football games, remains a significant challenge. To address this gap, we introduce Language-Controlled Diverse Style Policies (LCDSP), a novel LC-RL paradigm specifically designed for complex scenarios. LCDSP comprises two key components: a Diverse Style Training (DST) method and a Style Interpreter (SI). The DST method efficiently trains a single policy capable of exhibiting a wide range of diverse behaviors by modulating agent actions through style parameters (SP). The SI is designed to accurately and rapidly translate high-level language instructions into these corresponding SP. Through extensive experiments in a complex 5v5 football environment, we demonstrate that LCDSP effectively comprehends abstract tactical instructions and accurately executes the desired diverse behavioral styles, showcasing its potential for complex, real-world applications.

Project Page — https://lcdsp-webpage.github.io/LCDSP/

Introduction

Natural language (NL) serves as a powerful interface for humans to interact with and instruct AI agents. Training agents to follow NL instructions has been a long-standing goal in AI (Winograd 1972). Recent advancements in language-controlled reinforcement learning (LC-RL) (Luketina et al. 2019) have shown promising results, enabling agents to execute instructions in domains like navigation, object manipulation, and arrangement (Chen and Mooney 2011; Tellex et al. 2011; Hill et al. 2020; Brohan et al. 2023; Pang et al. 2023; Szot et al. 2024). Concurrently, significant progress has been made in training highly capable RL agents for complex, multi-agent environments such as Dota2 (Berner et al. 2019), StarCraft II (Vinyals et al. 2019), Gran Turismo (Wurman et al. 2022), and Google Research Football (GRF) (Song et al. 2024; Sun et al. 2024). However, a key challenge remains: effectively controlling these sophisticated policies in complex environments, particularly football games, to follow high-level, abstract instructions and exhibit specific, desired behavioral styles.

This challenge stems from two primary difficulties. First, human instructions in complex scenarios like football are often high-level and abstract, specifying not just a task outcome but also a desired **behavioral style** (e.g., "Prioritize defensive duties and perform a quick counterattack when opportunities arise."). Such instructions involve long-horizon planning, intricate coordination among multiple agents, and require modulating various aspects of agent behavior simultaneously, which is difficult for traditional LC-RL methods designed for simpler, task-oriented commands. Second, training LC-RL agents typically relies on evaluating whether an instruction has been successfully executed to provide training signals (e.g., through reward shaping process). While straightforward in simple tasks with clear completion criteria (Hill et al. 2020; Driess et al. 2023; Tan et al. 2024; Pang et al. 2023; Szot et al. 2024), determining the successful execution of abstract, style-based instructions in complex, dynamic environments like football is highly challenging and often lacks clear, rule-based criteria. Existing alternatives like human judgment (Brohan et al. 2023) or expert data (Fu et al. 2019; Bahdanau et al. 2019) are labor-intensive and have primarily been applied to simpler environments. Given these two challenges, developing methods that can interpret abstract instructions and enable policies to execute diverse behaviors in complex, multi-agent environments presents a significant research problem.

To address these two challenges, we propose a novel LC-RL paradigm, named Language-Controlled Diverse Style Policies (LCDSP). LCDSP empowers agents with diverse behavioral styles and enables fine-grained control over these styles through human instructions. Our approach involves training a single RL policy capable of exhibiting a wide spectrum of behaviors, modulated by continuous style parameters (SP). This is achieved through a novel Diverse Style Training (DST) method. Additionally, LCDSP incorporates a Style Interpreter (SI) module, specifically designed to accurately and rapidly translate high-level NL instructions into the corresponding SP. By training the policy to respond to SP rather than directly evaluating instruction completion, our method bypasses the difficulty of defining reward func-

[*]These authors contributed equally.
[†]Corresponding Author: Chen Chen

tions for abstract instructions during RL training. Furthermore, the SI module can be trained separately to perform the NL-to-SP mapping, offering a flexible and stable solution.

Our main contributions are summarized as follows: (1) We introduce LCDSP, a novel LC-RL paradigm that effectively comprehends high-level instructions and controls diverse policies in complicated, multi-agent environments, exemplified by football games; (2) We present a novel DST method and demonstrate its effectiveness in training a single policy capable of diverse behaviors in a complex environment involving multiple agents, cooperation, competition, and long-horizon planning. This marks the first application of multi-style training to such a complex simulation; (3) We propose the SI, a dedicated module designed for the accurate and rapid translation of abstract language instructions into SP. This provides agents with a practical and efficient means to comprehend high-level commands and operate effectively in complex environments.; (4) We conduct extensive experiments and policy evaluations, demonstrating the effectiveness, generality, and fine-grained control capabilities of our proposed LCDSP framework.

Related Works

Multi-Style RL Multi-style RL methods focus on training policies capable of exhibiting a range of distinct behavioral styles. These methods have found applications in various domains, including game AI (Mao et al. 2024; Mysore et al. 2022; Shen et al. 2020), robotic control (Abdolmaleki et al. 2020), and autonomous driving (Zhang et al. 2023). Multi-objective RL (MORL) is a related framework that can yield policies with varying behaviors (Abdolmaleki et al. 2020; Mossalam et al. 2016). The primary goal of MORL is to learn policies that optimize multiple competing objectives simultaneously. Some research aims to learn a set of policies approximating the Pareto frontier of optimal solutions (Zuluaga, Krause et al. 2016; Chen et al. 2019; Pirotta, Parisi, and Restelli 2015), while other approaches train a single policy using vectorized variants of standard RL algorithms (Basaklar, Gumussoy, and Ogras 2023). MORL typically seeks policies optimal under different linear combinations (weightings) of objectives. In contrast, our method develops behaviorally diverse policies by varying SP. These parameters serve as descriptors of desired behavioral characteristics and go beyond simple objective weightings, allowing for more nuanced control over policy execution. Multi-task RL (MTRL) is another related approach for generating policies with diverse capabilities (Lan et al. 2024; Liu et al. 2021). MTRL trains policies to complete a number of distinct tasks, where each task might implicitly require a specific behavioral style or skill. For instance, Yang et al. (2020) and He et al. (2024) employ routing networks to reconfigure a base policy for different tasks. However, MTRL is commonly applied to environments with a limited, predefined set of independent tasks, such as Meta-World (Yu et al. 2020). Our diverse style policies, enabled by the DST method, integrate diverse behaviors and are controlled by SP that allow for continuous variation and combinatorial expression of behaviors, enabling a much wider range of styles than a fixed set of discrete tasks.

Language-Conditioned RL Recent LC-RL approaches typically condition the policy on both the language instruction and the current observation, often by embedding both modalities (Jiang et al. 2019; Pang et al. 2023; Brohan et al. 2023; Hill et al. 2020; Song et al. 2023). Hill et al. (2020) encode NL instructions using BERT (Devlin et al. 2019) to condition the policy. TALAR (Pang et al. 2023) proposes learning a task-specific translator to convert NL to task. SayCan (Brohan et al. 2023) leverages large language models (LLMs) and grounds them through value functions to select language-conditioned skills. While effective in their respective domains, most prior LC-RL works have focused on interpreting and executing basic, specific instructions, primarily in simpler environments like object manipulation (Pang et al. 2023; Hill et al. 2020; Jiang et al. 2019) or navigation to a specific entity (Tellex et al. 2011). Although some methods combine these basic tasks to form longer sequences (Brohan et al. 2023; Song et al. 2023), they fundamentally remain sequential compositions of simple commands. In contrast, our method is designed to understand and execute high-level, abstract, style-based instructions in a complex, multi-agent environment by mapping these instructions to a diverse space of behavioral styles.

Preliminaries

RL is typically formulated as a Markov Decision Process (MDP) (Kumar et al. 2023; Sun et al. 2025), defined by the tuple $\langle S, A, P, r, \gamma \rangle$. Here, S represents the state space, and A denotes the action space. The transition function $P : S \times A \times S \to [0, 1]$ captures the environment dynamics, specifying the probability of transitioning to state $s_{t+1} \in S$ from state $s_t \in S$ by taking action $a \in A$. The reward function $r : S \times A \to R$ assigns a reward to each state-action pair. A policy $\pi(a|s)$ is the agent's behavior function, mapping states to actions or providing a probability distribution over actions. The value function $V^\pi(s)$ evaluates the quality of a state by predicting future rewards. In RL, the goal is to learn an optimal policy π^* that maximizes the expected discounted sum of rewards. Formally, the optimal policy is defined as: $\pi^* = \arg\max \mathbb{E}_s [V^\pi(s)]$, where the value function $V^\pi(s)$ is given by: $V^\pi(s) = \mathbb{E}_{\tau\sim\pi, P(s)} [\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. Here, $\gamma \in [0, 1)$ is the discount factor, and $\tau \sim \pi$ with $P(s)$ indicates sampling a trajectory τ for a horizon T starting from initial state s_0 using policy π, and $s_t \in \tau$ represents the state at t-th time step in the trajectory τ.

Conditioned RL can be formulated as an augmented MDP, which is defined by the tuple $\langle S, C, A, P, r_c, \gamma \rangle$. The additional tuple element C is the space of conditions, and other elements retain their definitions from the standard MDP. The reward function r_c: $S \times A \times C \to R$ assigns the reward to each state-action-condition triplet. Similarly, the policy $\pi(a|s, c)$ maps both states and conditions to actions. The objective in conditioned RL is to find a policy $\pi(a|s, c)$ that maximizes the expected discounted sum of rewards: $\mathbb{E}_{\tau\sim\pi, s_0\sim P(s_0), c\sim P_c} [\sum_{t=0}^{\infty} \gamma^t r_c(s_t, a_t, c)]$, where $P(c)$ represents a distribution over conditions in C, and $P(s_0)$ represents the distribution of initial states. This objective can also be expressed with a standard MDP by aug-

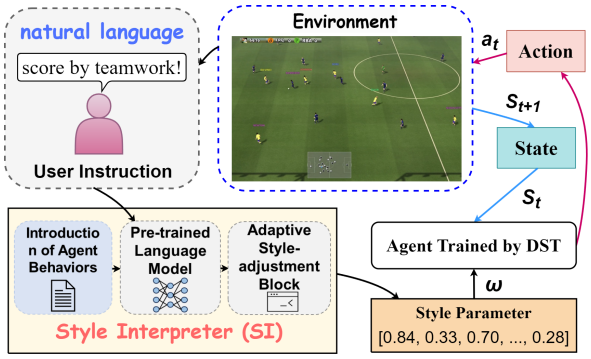

Figure 1: Overview of the inference process of LCDSP.

menting the state information with a condition context, explicitly conditioning the policy on c allows the agent to adapt its behavior based on different conditions.

Method

Overall Architecture

From a technical standpoint, achieving a method capable of understanding high-level human instructions and executing corresponding behaviors accurately and swiftly in complex environments necessitates overcoming two primary technical challenges. First, the development of a policy that can exhibit a wide range of diverse behaviors and is amenable to fine-grained control aligned with varied instructions. Second, the accurate and efficient translation of these instructions into a format directly consumable by the policy.

To address the first challenge, we propose the DST method. It enables the training of a single RL policy capable of fine-grained control over agent behaviors in complex environments through the SP. These parameters serve as continuous or discrete inputs that modulate the policy's actions. Furthermore, the DST method incorporates a novel sampling strategy to efficiently explore the style parameter space and accelerate the training process. To tackle the second challenge, we introduce the SI module. The SI is designed to accurately and rapidly translate high-level instructions into the corresponding SP that control the DST-trained policies.

The LCDSP paradigm integrates these two components, as illustrated in Figure 1. During inference, LCDSP receives a user instruction, then the SI processes this instruction to generate a corresponding SP vector, denoted as $\boldsymbol{\omega}$. This parameter vector $\boldsymbol{\omega}$, along with the current environment state s_t, is then fed into the trained policy, which outputs the appropriate action a_t to be executed in the environment, leading to the next state s_{t+1}. This modular design enables LCDSP to achieve its objective of guiding the RL policy to follow high-level instructions in complex environments.

Diverse Style Training (DST)

The DST method is designed to address the first key challenge described before. Similar to standard RL policy training, we begin by identifying the agent's main behaviors within the environment. We then implement reward shaping for those behaviors, allowing the agent's *preference* for

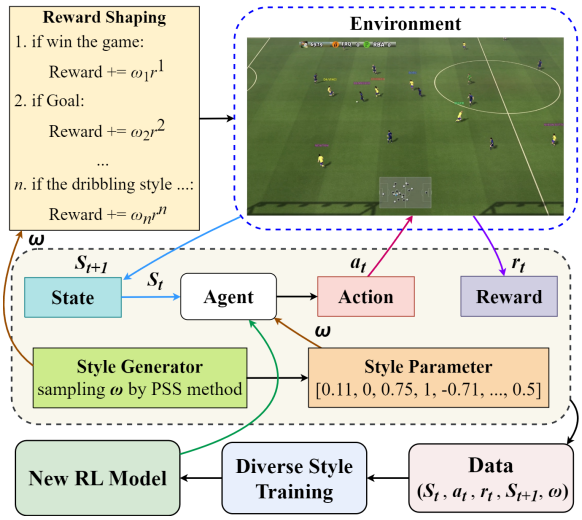

Figure 2: Overview of the DST process. For a given training scenario, we first design a reward shaping scheme for key agent behaviors, establishing how SP modulate the reward signal. Before each episode, the style generator samples a set of SP $\boldsymbol{\omega}$, which influence the rewards received for executing associated behaviors. The policy takes the current environment state s_t and SP $\boldsymbol{\omega}$ as input to produce action a_t. The environment returns the modulated reward r_t^{ω} and the next state s_{t+1}. These experiences are used by the DST process to update the RL policy.

a behavior to be modulated by a corresponding parameter within the reward function. The *preference* for a behavior, in this study, refers to the frequency or intensity with which the agent exhibits that behavior. A specific combination of preferences across various behaviors forms a *policy style*. The parameters used to control these behavioral preferences via the reward function are termed SP in this study. Specifically, SP are categorized into two types: *Float* and *Bool*. A *Float*-type parameter is continuous, ranging from 0 to 1, and can be mapped to different degrees or intensities of a behavior. A *Bool*-type parameter is binary, representing either *deactivate* or *activate* states for a behavior. We represent the SP as a vector $\boldsymbol{\omega} = [\omega_1, \omega_2, \ldots, \omega_n]$, where ω_i is the i-th parameter controlling the preference associated with the i-th behavior. Each unique vector $\boldsymbol{\omega}$ corresponds to a distinct behavioral style that the policy can potentially learn to exhibit. The vector $\boldsymbol{\omega}$ are crucial for achieving fine-grained control over the diverse styles within a single policy. During the DST process, $\boldsymbol{\omega}$ acts as a conditional input to the policy, as illustrated in Figure 2. This conditioning on $\boldsymbol{\omega}$ allows the policy to exhibit different behavioral styles even when presented with identical environment states. By varying $\boldsymbol{\omega}$, the trained policy learns to exhibit a wide range of behaviors and their combinations, enabling it to handle diverse situations.

The RL policy with SP can be optimized by policy gradient algorithms, the policy optimization criterion J_{π_θ} is proportional to the advantage function A^{π_θ}, as shown below:

$$J_{\pi_\theta} \propto \log\left(\pi_\theta(a \mid s, \omega)\right) A^{\pi_\theta}(s, \omega, a) \tag{1}$$

where $A^{\pi_\theta}(s,\omega,a) = Q^{\pi_\theta}(s,\omega,a) - V^{\pi_\theta}(s,\omega)$, the policy π is parameterized by θ, $V^{\pi_\theta}(s,\omega)$ is the value-function, and $Q^{\pi_\theta}(s,\omega,a)$ is the expected return of taking action a in state s under SP $\boldsymbol{\omega}$.

The value estimator V_ϕ, parameterized by ϕ, is optimized with optimization criteria $J_{V_\phi^\pi}$:

$$J_{V_\phi^\pi} \propto \| \left(V_\phi^\pi(s,\omega) - \left(r^\omega(s,\pi(s,\omega)) + V_\phi^\pi\left(s',\omega\right)\right) \| \quad (2)$$

where s' is the next state obtained from the environment after taking action a, and r^ω is the reward function modulated by SP $\boldsymbol{\omega}$.

As the complexity of target behaviors and environments increases, more SP are typically required to train policies that meet desired specifications. However, training difficulty escalates significantly as the number of SP grows. Current MORL methods typically support only 2 to 4 objectives (Felten et al. 2023), whereas our training scenario incorporates 10 agent behaviors. Training a single policy to exhibit distinct and controllable styles across diverse SP combinations is difficult, as many combinations might lead to ambiguous or ineffective behaviors. Furthermore, incorporating more behavioral styles significantly expands the exploration space. To address this challenge, we propose the Prioritized Style Sampling (PSS) method. PSS enhances training efficiency by adaptively adjusting the sampling probability of different SP during training. PSS prioritizes sampling SP that are expected to lead to more distinct behaviors, as measured by the entropy of policies across different styles. We compute the sampling distribution for each style parameter independently. For a continuous SP vector $\boldsymbol{\omega}$, the sampling probability $P(\omega_i)$ of the i-th parameter ω_i is defined as:

$$P\left(\omega_i\right) = \frac{f\left(H^i_{\max}(\omega) - H\left(\omega_i\right)\right)}{\int f\left(H^i_{\max}(\omega) - H\left(\omega_i\right)\right) d\omega_i} \quad (3)$$

where $H(\omega_i)$ is the expected action entropy when conditioned on ω_i, computed over the distribution of other SP combinations $\boldsymbol{\omega}_{-i}$, as defined below:

$$H(\omega_i) = \mathbb{E}_{\omega_{-i}|\omega_i, s\sim\rho^{\pi_\theta}}\left[-\sum_a \pi(a|s,\omega)\log\pi(a|s,\omega)\right] \quad (4)$$

And $H^i_{max}(\omega) = \max_{\omega_i\in\Omega_i} H(\omega_i)$, where Ω_i denotes the feasible range of the i-th SP. $f(x)$ is a monotonically increasing function, we use $f(x) = e^x$ in this study.

This formulation ensures that specific SP combinations with lower entropy are sampled with higher probability. It enables the model to focus more on combinations of SP that reduce entropy during training. The combinations of SP that reduce entropy indicate more effective learning. In contrast, SP combinations that do not reduce entropy through training suggest that these parameters may struggle to form effective policy styles. For discrete SP values, the sampling probabilities can be directly computed, which is equivalent to applying the Softmax to $H(\omega_i)$ when $f(x) = e^x$. For continuous SP values, when employing the PSS method, discretization into small intervals can be performed to facilitate the computation of entropy distribution statistics and subsequent sampling.

A style encoder is used to learn the representation of SP, which are then concatenated with the environment state and forwarded to the subsequent process. With the DST method, the trained policies are capable of exhibiting diverse styles with same environment state, and their behaviors can be finely adjusted through the corresponding SP.

Style Interpreter (SI)

When applying a trained diverse style policy, it is essential to select an appropriate behavioral style based on a given NL instruction. As described before, the behavioral style is controlled by a vector of SP $\boldsymbol{\omega}$. While LLMs show promise in language understanding, directly translating complex, high-level instructions into precise numerical SP values presents significant challenges. This process requires the model to not only comprehend the abstract meaning of the instruction but also to precisely map it to specific parameter values (which may be continuous or binary), all while maintaining an acceptable inference time for practical deployment.

To address these challenges, we propose the Style Interpreter (SI) module, specifically designed to accurately and rapidly translate high-level human instructions into their related SP values. Furthermore, the SI can effectively leverage the information about the specific behaviors controlled by each parameter for accurate translation. Specifically, SI is implemented by fine-tuning a network built upon a pre-trained language model (PLM). A user instruction is first fed into the PLM to obtain its representation, which then serves as the primary input for an Adaptive Style-adjustment Block (ASaB) responsible for generating the SP. As the key component of the SI, ASaB facilitates this translation by incorporating information about the specific behaviors controlled by each SP. In our scenario, each main agent behavior is associated with a fixed natural language "description" or "introduction", and these behavior introductions are also processed by the same PLM to obtain their fixed representations. As shown in Figure 3, the user instruction representation is fed into a network that produces initial outputs (logits) Y. Simultaneously, for each style parameter i, the representation of its corresponding behavior introduction is fed into the ASaB to generate two adaptive scaling parameters, γ_i and β_i. The ASaB then applies an affine transformation using these scaling parameters to the initial output Y for obtaining the final SP:

$$\text{ASaB}(Y_i \mid \gamma_i, \beta_i) = \gamma_i \cdot Y_i + \beta_i \quad (5)$$

Here, Y_i represents the initial output for the i-th SP, derived from the representation of user instruction. γ_i and β_i are the learned scale and bias parameters for the i-th SP, generated based on the representation of the i-th behavior introduction. The network incorporates multiple output heads, one for each SP, and the overall training objective is a multi-head regression task to predict the ground truth SP $\boldsymbol{\omega}$.

This adaptive scaling mechanism, conditioned on the behavior introductions, allows the SI to dynamically adjust the translation for each style parameter based on the specific behavior it controls, guided by the overall instruction. This enhances the SI module's ability to capture the nuanced relationship between user instructions, behaviors, and desired

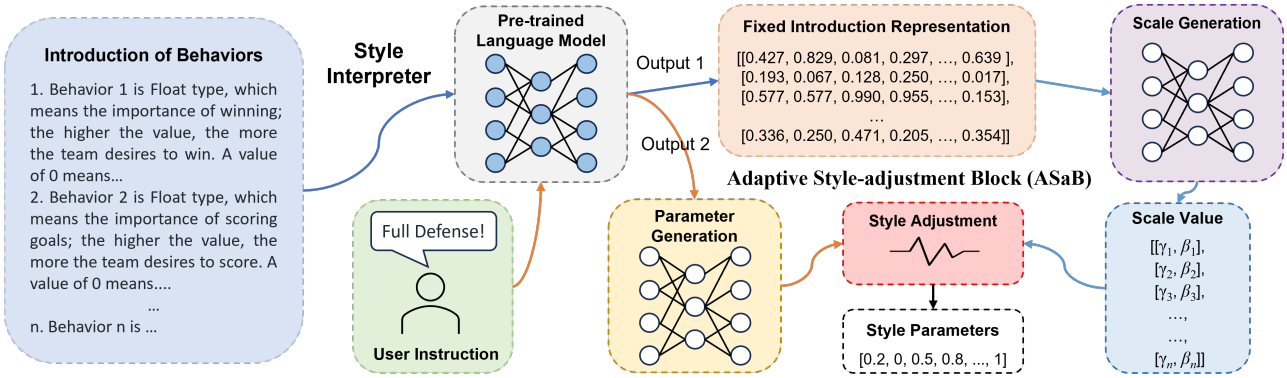

Figure 3: Overview of the SI module. User instructions are input into a PLM to obtain their representations. These instruction representations are then processed by an additional network to output logits. Concurrently, the introductions of main agent behaviors are also processed by the same PLM to obtain their representations. These behavior representations are fed into the ASaB to generate adaptive scaling parameters (γ_i, β_i) for each style parameter i. Finally, the SP are generated by applying the ASaB transformation to the initial outputs to get the final SP.

SP values, leading to improved accuracy and stability in the translation process. With the SI, the LCDSP is capable of aligning complex human instructions with appropriate SP to control the diverse style policy effectively and rapidly.

Experiments

To systematically evaluate the LCDSP, we conducted comprehensive experiments across three aspects. First, we assessed LCDSP's integrated ability in executing high-level instructions within a complex scenario. Second, we demonstrate the performance of DST through analyzing the training efficiency and examining their generalization ability. Finally, we investigated the instruction translation performance by comparing our proposed SI with several baselines.

High-Level Instructions Following Performance

Environment The 5v5 scenario within GRF is an open-source environment designed to support long-horizon and complex tasks (Sun et al. 2024). Each episode consists of 3K steps and features a sparse reward structure. It also incorporates rules analogous to those in realistic football. This challenging task requires the simultaneous control of five agents and necessitates intricate multi-agent cooperation. Detailed information about the scenario and input/output design can be found in Appendices A.1 and A.2, respectively. We identified ten key agent behaviors for training the diverse style policy: the preference of *Win*, *Goal*, *Lose Goal*, *Hold Ball*, *Get Possession*, *Pass*, *Spacing*, *Shot*, *Move*, and *Formation*. Accordingly, ten SP and their corresponding reward functions are employed, which are detailed in Appendix A.3. The diverse style policy is initially trained by competing against a built-in AI and subsequently improved through a self-play training approach, leveraging the adversarial nature of the environment. More details regarding the DST training process are provided in Appendices A.4-A.6.

Instructions To demonstrate that the LCDSP can understand and follow high-level abstract instructions, we designed six tactics analogous to real-world football strategies: *Positive Attack*, *All-out Attack*, *Balanced Play*, *Counter Attack*, *Park the Bus*, and *Tiki-Taka*. For each tactic, we tested 30 instructions, each accompanied by a sentence elucidating the corresponding strategy. To establish baselines for comparison, we utilized three popular LLMs: GPT-4o (OpenAI 2024), Claude 3.5-Sonnet (Anthropic 2024), and Llama 3.1-8B (Dubey, Jauhri, and et al. 2024). These LLMs were used to align SP with those instructions, serving as alternatives to our SI module. Detailed information on the employed instructions and the SP aligning process by LLMs can be found in Appendices B.1 and B.2, respectively.

Results In this complex scenario, it is impractical to use predefined rules to determine the successful execution of high-level instructions. Therefore, we utilized in-game metrics to evaluate the extent to which LCDSP adheres to the specified tactic instructions, as illustrated in Figure 4. The results indicate that our method accurately comprehends high-level instructions and executes appropriate behavioral styles. Furthermore, it demonstrates distinct performances under different tactical instructions and exhibits good generalization across various instruction translation approaches. It showcasing the wide variety of policies that can be controlled by NL and effectively resemble real-world tactics. Performance across our method and different baselines is generally consistent, though variations in the degree of behavior are observed in certain cases. For example, our method demonstrates superior style similarity, albeit at the cost of a lower win rate compared to most baselines. Additionally, Claude3.5 tends to have a conservative policy style in *Positive Attack*, while Llama3.1 exhibits more passing attempts in *Tiki-Taka*. Two examples of rendered frames during the execution of *Counter Attack* and *Tiki-Taka* are provided in Appendix C.

Training Performance of Diverse Style Policies

During the training process of DST, the style generator utilizes the proposed PSS method to produce SP. As mentioned in Section 4.2, the PSS is designed to improve the DST per-

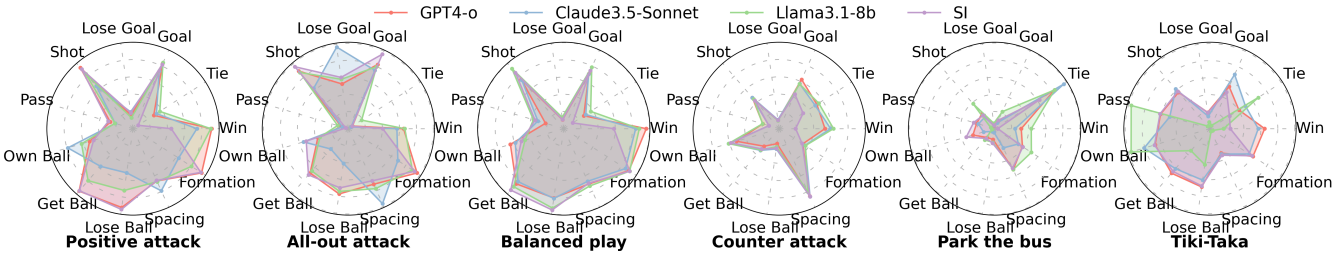

Figure 4: Comparison of in-game metrics under different tactical instructions. The tests were conducted using self-play matches, where the opponents' SP were randomly generated, and each instruction were tested ten times. Both the *Positive Attack* and *All-out Attack* tactics result in a higher number of goals and shot attempts compared to the *Balanced Play* tactic. However, the *All-out Attack* tactic exhibits a lower win rate due to a higher number of conceded goals and fewer draws, attributable to its overly aggressive style. The *Counter Attack* tactic shows a higher draw rate; to facilitate counter-attacks, the formation is positioned deeper, creating a larger space between forwards and defenders. The *Park the Bus* tactic features more compact spacing, leading to a very high draw rate. The *Tiki-Taka* tactic achieves the highest possession ratio and number of pass attempts, with closer spacing facilitating short passes.

formance, particularly when dealing with a vast combination of SP. To evaluate this improvement, we assessed it from two perspectives: training efficiency and policy generalization.

Baselines We compare PSS with two parameter sampling approaches commonly used in RL policy training as baselines. **Uniform** sampling is a simple popular method that maintains a static uniform distribution over the SP space throughout training. **LSDR** (Mozian et al. 2020) is a method that requires a predefined reference style parameter distribution and learns a multivariate Gaussian training distribution to maximize generalization to this reference. Unlike our PSS method that use entropy-based criteria, LSDR gives priority to the SP where the current policy can achieve high returns. The implementations of those baselines are detailed in A.8.

Training Efficiency To obtain quantitative results for the improvement in training efficiency achieved by the PSS method, we conducted a comparative analysis of the PSS against these baselines. Table 1 presents the comparison results for three key indicators: Style-based Expected Utility (SEU), Style-based Maximum Utility Loss (SMUL), and Style-based ELO rating (SELO). These indicators are often used in MTRL or MORL studies, which are detailed in Appendix A.7. The results indicate that the PSS method leads to higher training efficiency, which demonstrates superior performance in both SEU and SMUL metrics, indicating greater expected returns across various style parameter distributions and improved training efficiency. We also evaluated these trained models against their combined historical model pool to calculate the SELO scores. PSS achieves a higher SELO score, indicating that in adversarial environments, the PSS method not only expands the policy's range of styles but also enhances the model's strength more rapidly.

Policy Generalization To demonstrate that our proposed method exhibits excellent generalization for style policies, we finely adjusted the SP and recorded their corresponding in-game metrics. We used four *Float*-type SP for this evaluation. For each target SP, we fixed the remaining SP at their baseline values and uniformly sampled 20 values from

Metrics	SEU (↑)	SMUL (↓)	SELO (↑)
Uniform	1.81 ± 0.09	4.13 ± 0.12	1029 ± 8.32
LSDR	1.87 ± 0.09	4.08 ± 0.17	1048 ± 9.57
PSS	**3.50 ± 0.03**	**2.44 ± 0.07**	**1142 ± 10.27**

Table 1: Comparison of training efficiency for the PSS method against baselines.

0 to 1 in increments of 0.05. For each sampled value, we ran 1K episodes against the same opponent model with random styles. In addition, we conducted a comparison of the PSS model against these two baselines. The variations in in-game metrics resulting from fine-grained adjustments of these SP are shown in Figure 5. It can be observed that adjusting the value of each style parameter results in a nearly linear and smooth change in the corresponding metric. It indicates that the policies trained with PSS possess the property of smooth linear interpolation, which is beneficial for generalization. Furthermore, the PSS-trained model exhibits wider ranges of adjustment for the in-game metrics compared to the baselines, suggesting a broader coverage of the style space.

More tests and analysis about the training efficiency and policy generalization can be found in Appendix A.9.

Performance of Style Interpreter

The SI is a pivotal module within our LCDSP method, responsible for rapidly converting high-level, abstract instructions into SP that control the trained policy. To quantitatively assess the instruction comprehension capabilities of SI, we designed a dedicated test, described below.

Dataset We tested the SI on a dataset of 8.6K high-level instructions. We generated 1.4K instructions for each tactic, and each instruction is a sentence elucidating the corresponding tactic. A rigorous process was employed to label the corresponding SP for each instruction. Detailed information regarding the instruction generation and SP labeling processes can be found in Appendices B.1 and B.2, respec-

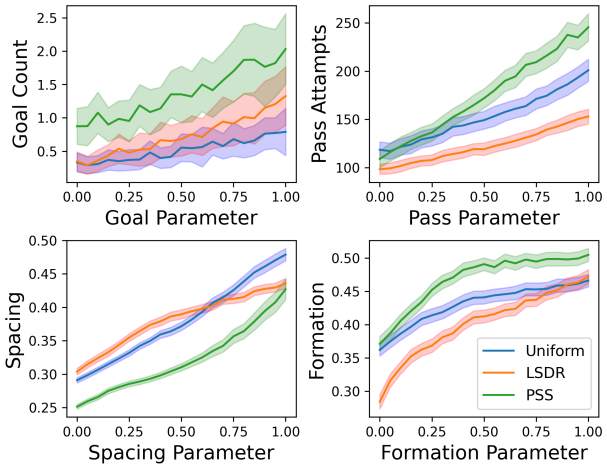

Figure 5: Fine-grained adjustment of SP and their corresponding changes in in-game metrics for different methods.

tively.

Configuration We utilize Qwen2.5-0.5B (Alibaba 2024) as the PLM within the SI module. Qwen2.5-0.5B is an open-source LLM with a relatively small number of parameters. The translation process is modeled as a multi-head regression task, where the label for each instruction is the set of SP corresponding to the key behaviors. The introduction of SP in the SI is to define their types and the key behaviors they control within the scenario. During the training, the parameters of PLM are frozen, and its last hidden state is used as the representation. Eighty and twenty percent of the instructions were used for training and validation, respectively.

Baselines We compare SI with three decent language comprehension modules from popular language-conditioned policy learning methods as baselines. **Hill et al.** (Hill et al. 2020) proposed a method for training RL agents with BERT to follow human instructions, employing various methods for information fusion. **BC-Z** (Jang et al. 2022) uses a FiLM layer to condition on the language instruction to guide multi-head action prediction for diverse manipulation tasks. **TALAR** (Pang et al. 2023) directly fine-tunes BERT as a translator to encode instructions into inputs for RL policies without incorporating other modalities.

Results We use the Mean Absolute Error (MAE) and inference time to evaluate the instruction-to-parameter translation capability of our SI module. MAE reflects the ability to match the translated parameters with the ground truth SP. The inference time of the translation process is also an important consideration, as faster inference is crucial for practical application. Table 2 shows the MAE and inference time for each method. Our proposed SI outperforms the three baselines in terms of MAE. TALAR exhibits the shortest inference time due to its simpler network structure. The MAE curves of validation set for our method compared to baselines during training are shown in Figure 6 (a).

Ablation studies were conducted to estimate the individual contributions of the critical components within SI. While

Methods	MAE (↓)	Inference time (s) (↓)
Comparisons with Baselines		
Hill et al.	0.923 ± 0.023	0.032 ± 0.024
BC-Z	0.751 ± 0.026	0.129 ± 0.090
TALAR	0.992 ± 0.021	**0.009 ± 0.025**
SI (Our method)	**0.671 ± 0.015**	0.094 ± 0.027
Ablation Test		
without ASaB	0.752 ± 0.014	0.023 ± 0.028
with matmul fusion	0.846 ± 0.064	0.125 ± 0.028
with cross-attention	1.477 ± 0.054	0.033 ± 0.026
with Qwen2.5-3B	0.629 ± 0.005	0.785 ± 0.024

Table 2: Comparison of MAE and inference time among baselines and ablation tests.

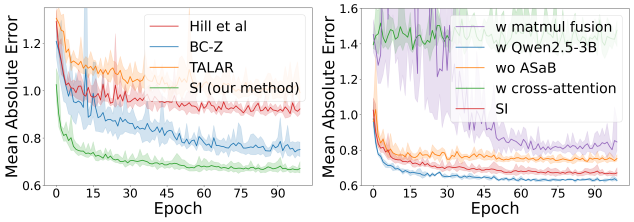

Figure 6: The curves of validation error for our method compared to baselines and ablation configurations.

the quality of the PLM is important for translation ability, the effectiveness of our method substantially benefits from the ASaB approach. Furthermore, alternative structures, specifically a matrix multiply fusion and a cross-attention structure, were also evaluated against our ASaB approach. In addition, we evaluated the performance using the same PLM but with a larger parameter size (Qwen2.5-3B). The outcomes, shown in Table 2, demonstrate that the ASaB enhances instruction comprehension performance. The cross-attention structure failed to converge within a limited training epochs, which may be due to its complexity. The larger model (Qwen2.5-3B) yields better results, but its size and inference time are several times greater than the original configuration. The MAE curves of validation set for the ablation tests are given in Figure 6 (b). The implementation details of our SI module, baselines, and ablations are provided in Appendix B.3.

Conclusion

We introduce Language-Controlled Diverse Style Policies, a novel language-controlled reinforcement learning paradigm that equips agents with diverse style policies, enabling fine-grained control through high-level instructions in complex multi-agent environments like football games. The RL policy is trained using a specially designed Diverse Style Training method, and the crucial alignment between instructions and policies is achieved via a novel Style Interpreter module. Extensive experiment results demonstrate the effectiveness and generality of our proposed method, thereby confirming its capability to execute abstract, high-level instructions in complex environments.

Acknowledgments

We would like to express our sincere gratitude to the reviewers for their insightful comments and valuable suggestions. We also thank our colleagues at the Sports Products Department, Interactive Entertainment Group, Tencent for their helpful discussions and support throughout this work.

References

Abdolmaleki, A.; Huang, S.; Hasenclever, L.; Neunert, M.; Song, F.; Zambelli, M.; Martins, M.; Heess, N.; Hadsell, R.; and Riedmiller, M. 2020. A distributional view on multi-objective policy optimization. In *International conference on machine learning*, 11–22. PMLR.

Alibaba. 2024. Qwen2.5: A Party of Foundation Models. https://qwenlm.github.io/blog/qwen2.5/. Accessed: 2024-09-19.

Anthropic. 2024. Claude 3.5 Sonnet. https://www.anthropic.com/news/claude-3-5-sonnet.

Bahdanau, D.; Hill, F.; Leike, J.; Hughes, E.; Kohli, P.; and Grefenstette, E. 2019. Learning to Understand Goal Specifications by Modelling Reward. In *International Conference on Learning Representations*.

Basaklar, T.; Gumussoy, S.; and Ogras, U. 2023. PD-MORL: Preference-Driven Multi-Objective Reinforcement Learning Algorithm. In *The Eleventh International Conference on Learning Representations*.

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Dkebiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Brohan, A.; Chebotar, Y.; Finn, C.; Hausman, K.; Herzog, A.; Ho, D.; Ibarz, J.; Irpan, A.; Jang, E.; Julian, R.; et al. 2023. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, 287–318. PMLR.

Chen, D.; and Mooney, R. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, 859–865.

Chen, X.; Ghadirzadeh, A.; Björkman, M.; and Jensfelt, P. 2019. Meta-learning for multi-objective reinforcement learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 977–983. IEEE.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Burstein, J.; Doran, C.; and Solorio, T., eds., *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, 4171–4186. Association for Computational Linguistics.

Driess, D.; Xia, F.; Sajjadi, M. S. M.; Lynch, C.; Chowdhery, A.; Ichter, B.; Wahid, A.; Tompson, J.; Vuong, Q.; Yu, T.; Huang, W.; Chebotar, Y.; Sermanet, P.; Duckworth, D.; Levine, S.; Vanhoucke, V.; Hausman, K.; Toussaint, M.; Greff, K.; Zeng, A.; Mordatch, I.; and Florence, P. 2023. PaLM-E: an embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Dubey, A.; Jauhri, A.; and et al., A. P. 2024. The Llama 3 Herd of Models. arXiv:2407.21783.

Felten, F.; Alegre, L. N.; Nowe, A.; Bazzan, A. L. C.; Talbi, E. G.; Danoy, G.; and da Silva, B. C. 2023. A Toolkit for Reliable Benchmarking and Research in Multi-Objective Reinforcement Learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Fu, J.; Korattikara, A.; Levine, S.; and Guadarrama, S. 2019. From Language to Goals: Inverse Reinforcement Learning for Vision-Based Instruction Following. In *International Conference on Learning Representations*.

He, J.; Li, K.; Zang, Y.; Fu, H.; Fu, Q.; Xing, J.; and Cheng, J. 2024. Not All Tasks Are Equally Difficult: Multi-Task Deep Reinforcement Learning with Dynamic Depth Routing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 12376–12384.

Hill, F.; Mokra, S.; Wong, N.; and Harley, T. 2020. Human instruction-following with deep reinforcement learning via transfer-learning from text. *arXiv preprint arXiv:2005.09382*.

Jang, E.; Irpan, A.; Khansari, M.; Kappler, D.; Ebert, F.; Lynch, C.; Levine, S.; and Finn, C. 2022. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, 991–1002. PMLR.

Jiang, Y.; Gu, S. S.; Murphy, K. P.; and Finn, C. 2019. Language as an abstraction for hierarchical deep reinforcement learning. *Advances in Neural Information Processing Systems*, 32.

Kumar, N.; Derman, E.; Geist, M.; Levy, K. Y.; and Mannor, S. 2023. Policy gradient for rectangular robust markov decision processes. *Advances in Neural Information Processing Systems*, 36: 59477–59501.

Lan, S.; Zhang, R.; Yi, Q.; Guo, J.; Peng, S.; Gao, Y.; Wu, F.; Chen, R.; Du, Z.; Hu, X.; et al. 2024. Contrastive modules with temporal attention for multi-task reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Liu, B.; Liu, X.; Jin, X.; Stone, P.; and Liu, Q. 2021. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34: 18878–18890.

Luketina, J.; Nardelli, N.; Farquhar, G.; Foerster, J.; Andreas, J.; Grefenstette, E.; Whiteson, S.; and Rocktäschel, T. 2019. A Survey of Reinforcement Learning Informed by Natural Language. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 6309–6317. International Joint Conferences on Artificial Intelligence Organization.

Mao, Y.; Wu, C.; Chen, X.; Hu, H.; Jiang, J.; Zhou, T.; Lv, T.; Fan, C.; Hu, Z.; Wu, Y.; Hu, Y.; and Zhang, C. 2024. Stylized Offline Reinforcement Learning: Extracting Diverse High-Quality Behaviors from Heterogeneous Datasets. In *The*

Twelfth International Conference on Learning Representations.

Mossalam, H.; Assael, Y. M.; Roijers, D. M.; and Whiteson, S. 2016. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*.

Mozian, M.; Camilo Gamboa Higuera, J.; Meger, D.; and Dudek, G. 2020. Learning Domain Randomization Distributions for Training Robust Locomotion Policies. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 6112–6117.

Mysore, S.; Cheng, G.; Zhao, Y.; Saenko, K.; and Wu, M. 2022. Multi-Critic Actor Learning: Teaching RL Policies to Act with Style. In *International Conference on Learning Representations*.

OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774.

Pang, J.-C.; Yang, X.-Y.; Yang, S.-H.; Chen, X.-H.; and Yu, Y. 2023. Natural language instruction-following with task-related language development and translation. *Advances in Neural Information Processing Systems*, 36.

Pirotta, M.; Parisi, S.; and Restelli, M. 2015. Multi-objective reinforcement learning with continuous pareto frontier approximation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Shen, R.; Zheng, Y.; Hao, J.; Meng, Z.; Chen, Y.; Fan, C.; and Liu, Y. 2020. Generating Behavior-Diverse Game AIs with Evolutionary Multi-Objective Deep Reinforcement Learning. In *IJCAI*, 3371–3377.

Song, C. H.; Wu, J.; Washington, C.; Sadler, B. M.; Chao, W.-L.; and Su, Y. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2998–3009.

Song, Y.; Jiang, H.; Tian, Z.; Zhang, H.; Zhang, Y.; Zhu, J.; Dai, Z.; Zhang, W.; and Wang, J. 2024. An Empirical Study on Google Research Football Multi-agent Scenarios. *Machine Intelligence Research*, 1–22.

Sun, C.; Shen, S.; Tao, W.; Xue, D.; and Zhou, Z. 2025. Noise-Resilient Symbolic Regression with Dynamic Gating Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Sun, C.; Shen, S.; Xue, D.; Tao, W.; and Zhou, Z. 2024. Enhancing AI-Bot Strength and Strategy Diversity in Adversarial Games: A Novel Deep Reinforcement Learning Framework. *IEEE Transactions on Games*.

Szot, A.; Schwarzer, M.; Agrawal, H.; Mazoure, B.; Metcalf, R.; Talbott, W.; Mackraz, N.; Hjelm, R. D.; and Toshev, A. T. 2024. Large Language Models as Generalizable Policies for Embodied Tasks. In *The Twelfth International Conference on Learning Representations*.

Tan, W.; Zhang, W.; Liu, S.; Zheng, L.; Wang, X.; and An, B. 2024. True Knowledge Comes from Practice: Aligning Large Language Models with Embodied Environments via Reinforcement Learning. In *The Twelfth International Conference on Learning Representations*.

Tellex, S.; Kollar, T.; Dickerson, S.; Walter, M.; Banerjee, A.; Teller, S.; and Roy, N. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 25, 1507–1514.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; Oh, J.; Horgan, D.; Kroiss, M.; Danihelka, I.; Huang, A.; Sifre, L.; Cai, T.; Agapiou, J. P.; Jaderberg, M.; Vezhnevets, A. S.; Leblond, R.; Pohlen, T.; Dalibard, V.; Budden, D.; Sulsky, Y.; Molloy, J.; Paine, T. L.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Wu, Y.; Ring, R.; Yogatama, D.; Wünsch, D.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T.; Kavukcuoglu, K.; Hassabis, D.; Apps, C.; and Silver, D. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Winograd, T. 1972. Understanding natural language. *Cognitive Psychology*, 3(1): 1–191.

Wurman, P. R.; Barrett, S.; Kawamoto, K.; MacGlashan, J.; Subramanian, K.; Walsh, T. J.; Capobianco, R.; Devlic, A.; Eckert, F.; Fuchs, F.; Gilpin, L.; Khandelwal, P.; Kompella, V.; Lin, H.; MacAlpine, P.; Oller, D.; Seno, T.; Sherstan, C.; Thomure, M. D.; Aghabozorgi, H.; Barrett, L.; Douglas, R.; Whitehead, D.; Dürr, P.; Stone, P.; Spranger, M.; and Kitano, H. 2022. Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature*, 602(7896): 223–228.

Yang, R.; Xu, H.; Wu, Y.; and Wang, X. 2020. Multi-task reinforcement learning with soft modularization. *Advances in Neural Information Processing Systems*, 33: 4767–4777.

Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; and Levine, S. 2020. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, 1094–1100. PMLR.

Zhang, H.; Lin, Y.; Han, S.; and Lv, K. 2023. Lexicographic Actor-Critic Deep Reinforcement Learning for Urban Autonomous Driving. *IEEE Transactions on Vehicular Technology*, 72(4): 4308–4319.

Zuluaga, M.; Krause, A.; et al. 2016. e-pal: An active learning approach to the multi-objective optimization problem. *Journal of Machine Learning Research*, 17(104): 1–32.