LARP: Language-Agent Role Play for Open-World Games

Ming Yan [*†1], **Ruihao Li** [*1], **Hao Zhang**[1], **Hao Wang**[1], **Zhilan Yang**[1], **and Ji Yan**[1]
[1]MiAO

Abstract

Language agents have shown impressive problem-solving skills within defined settings and brief timelines. Yet, with the ever-evolving complexities of open-world simulations, there's a pressing need for agents that can flexibly adapt to complex environments and consistently maintain a long-term memory to ensure coherent actions. To bridge the gap between language agents and open-world games, we introduce Language Agent for Role-Playing (LARP), which includes a cognitive architecture that encompasses memory processing and a decision-making assistant, an environment interaction module with a feedback-driven learnable action space, and a postprocessing method that promotes the alignment of various personalities. The LARP framework refines interactions between users and agents, predefined with unique backgrounds and personalities, ultimately enhancing the gaming experience in open-world contexts. Furthermore, it highlights the diverse uses of language models in a range of areas such as entertainment, education, and various simulation scenarios. The project page is released at `https://miao-ai-lab.github.io/LARP/`.

1 Introduction

Large Language Models (LLMs) are machine learning models capable of performing a variety of Natural Language Processing (NLP) tasks such as generating text, translating text from one language to another, and answering questions conversationally. The term *large* refers to the vast amount of parameters the language model can update during its learning process. With the development of pre-training generative model techniques and the construction of massive and comprehensive datasets, some top-performing large language models have up to hundreds of billions of parameters [Touvron et al., 2023, Radford et al., 2018, 2019, Brown et al., 2020, Ouyang et al., 2022, OpenAI, 2023]. Furthermore, owing to the advancements in large language models, AI entities have become a hot topic in recent years. These artificial intelligence entities, often referred to as *agents* [Russell and Norvig, 2010, Wooldridge and Jennings, 1995], are the fundamental components of large-scale artificial intelligence systems. Typically, in the realm of Artificial General Intelligence, an agent is an artificial entity that can perceive its surrounding environment through sensors, make decisions, and respond via actuators. With the development of large language models and agents, there is a new trend of combining them into a single entity called language agents. These language agents are designed by integrating large language models and agent design [Wang et al., 2023a, Xi et al., 2023, Sumers et al., 2023].

As a strongly related industry to computers, gaming has become increasingly intertwined with the development of general-purpose language agents. The application of LLMs and agents has grown more widespread. In related studies, there is considerable literature on the application of language agents in text-based games [Dambekodi et al., 2020, Singh et al., 2021, Yao et al., 2021, Urbanek et al., 2019] and adversarial games [OpenAI et al., 2019, Arulkumaran et al., 2019]. Concurrently, with the enhanced capabilities of LLMs, open-world games have emerged as the frontier for language agent applications. This is due to the unique and challenging scenarios present in open-world games, which provides a fertile ground for general-purpose language agents. Open-world games present a rich, dynamic, and engaging environment, encompassing complex missions and storylines. They require the use of agents to equip non-player characters with diversified behaviors. Although numerous studies have proposed architectures for general

[*]Equal Contribution
[†]Correspondence to: mingyan@miao.company

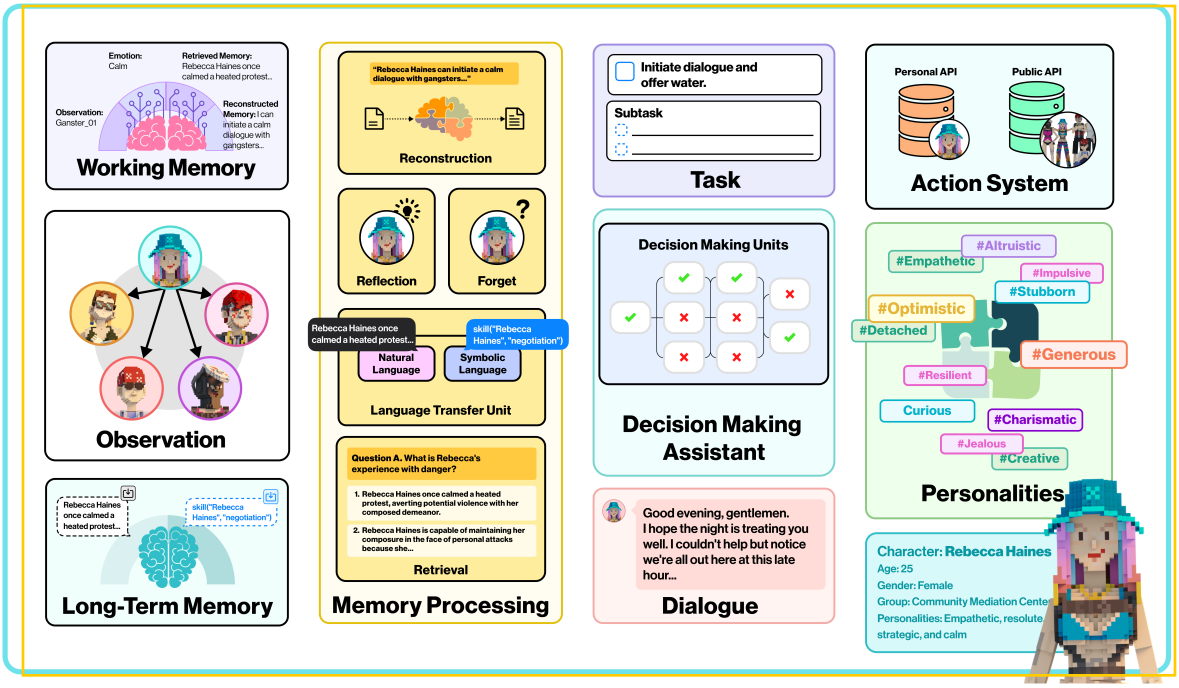

Figure 1: Cognitive Architecture of LARP Overview.

language agents that could be applied in open-world games like Minecraft [Lin et al., 2023, Park et al., 2023], a gap still exists between the general-purpose agents and the overall requirements in an open-world gaming context. General-purpose language agents are created to address a variety of issues in realistic environments. Their primary requisites are universality and the simulation of human behavior. These agents can adapt to various environments and tasks, without being restricted to fixed roles. However, these general-purpose language agents face significant challenges in practical open-world environments. These challenges include but are not limited to interpreting complex environments, memorizing long-term events, generating expressions that cohere with character and environmental settings, and continuously learning from interactions with the environment.

Hence, in this work, we propose a game-oriented Role-Playing Agent framework, Language Agent for Role Play (LARP) toward open-world games. LARP focuses on blending the open-world games with language agents, utilizing a modular approach for memory processing, decision-making, and continuous learning from interactions. In the agent's internal depiction, we designed a complex cognitive architecture based on cognitive psychology, equipping agents under the LARP framework with high playability and uniquity. Aiming to yield more realistic role-playing experience, we regularized agents using the data and context of the open-world gaming environment, prior set personalities, knowledge, rules, memory, and post constraints, which can be seen as a specific case within the general-purpose language agents. As for the general agent architecture, it typically requires a large-scale language model. However, our architecture incorporates a cluster of smaller language models, each fine-tuned for different domains, to handle various tasks separately. This design contributes new experiences and perspectives to developing language agents for open-world role-playing games.

The rest of this work is organized as follows: Section 2 discusses the related works on agent framework and agent components. In Section 3, we present the cognitive architecture part of LARP. In Section 4, we highlight the environmental interaction module, and in Section 5, we introduce agents' alignment of diversified personalities. A discussion is presented in Section 6, followed by a conclusion of the entire paper in Section 7.

2 Related Work

The development and advancements in LLMs have brought opportunities and potential to various fields. These fields include but are not limited to, role-playing, gaming, and tool-using Agents, amongst others. At the same time, the definition of language agents is being continuously updated. We will share some related works on agent architecture in this section.

2.1 Agent Framework

Firstly, we will introduce some works related to language agents' role-playing and simulation, which are aimed at enhancing the LLM's ability in role-playing and highlighting distinct features. These works also aim to boost the interaction capability between the agents and users, making the agents appear more *self-conscious* [Wang et al., 2023b, Shao et al., 2023, Shanahan et al., 2023, Li et al., 2023a]. Other studies focus on role-playing and interaction among multiple agents, which include scenarios such as collaboration to complete tasks [Li et al., 2023b, Chen et al., 2023a, Qian et al., 2023, Li et al., 2023c, Wu et al., 2023], simulating daily activities [Lin et al., 2023, Park et al., 2023, Wang et al., 2023c, Liu et al., 2023a], and promoting progress in debates [Liang et al., 2023, Du et al., 2023, Chan et al., 2023].

In addition to this, language agents are also applied in open-world environments. There are not only instances of application in text-based games [Urbanek et al., 2019, Côté et al., 2019, Hausknecht et al., 2020]but also exploration tasks in open-world environments such as Minecraft [Wang et al., 2023d,e, Zhu et al., 2023].

2.2 Agent Component

In this section, we will introduce some related works on the design of language agent components. An agent system is usually divided into three parts: memory, planning, and action (tool use) [Weng, 2023]. Memory system serves as a repository for facts, reflections, etc., entailing abilities for storage and retrieval. Hence, efforts in memory primarily pertain to input/output functions, including memory compression [Hu et al., 2023], storage, and retrieval [Park et al., 2023, Zhong et al., 2023, Huang et al., 2023].

The planning component is responsible for the decision-making aspect related to the agent's behavior and language. The capability of agents largely depends on this part. Abilities for planning [Yao et al., 2023, Liu et al., 2023b, Yao et al., 2022, Shinn et al., 2023, Liu et al., 2023c, Wang et al., 2023f] and reasoning [Wei et al., 2022, Madaan et al., 2023] are realized in this component, and associated works typically revolve around these two abilities.

The last component is tool usage and actions, which signifies an augmentation in the capabilities of the intelligent entities, aiding them in conducting more complex and difficult tasks. Works in this section include tool-using [Nakano et al., 2021] and learning new actions [Schick et al., 2023].

3 Cognitive Architecture

Cognitive architecture is a fundamental component of role-playing language agents in open-world games. It provides a logical framework and enables self-recognition of agents. The cognitive architecture is shown in figure 2. It comprises four major modules: long-term memory, working memory, memory processing, and decision-making. The long-term memory module serves as the primary warehouse containing memories with substantial storage capacity. Working memory acts as a temporary cache with limited space for memory. The memory processing module is the most important unit of the cognitive architecture. The decision-making module then derives agents' subsequent actions based on the retrieved information.

3.1 Long-Term Memory

In cognitive science, long-term memory (LTM) is comprised of two types of memory: declarative memory and procedural memory. Declarative memory is further divided into semantic memory and episodic memory [Laird,

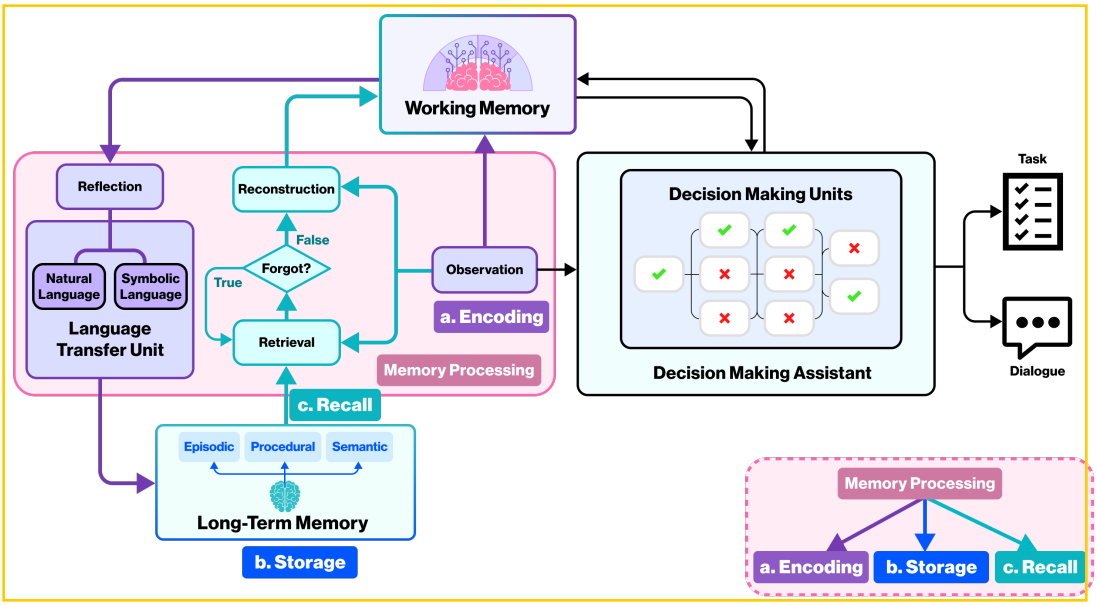

Figure 2: Cognitive Workflow of LARP. This represents a cycle: Information from long-term memory and observation is processed in the memory processing module and transmitted to the working memory module. The information in the working memory module, together with the observed information, is inputted into the decision-making assistant, which finally generates a decision or dialogue. Memory processing has three main stages: encoding, storage, and recall. Encoding is the process of transforming information into a form that can be stored in memory. Storage is the process of maintaining information in memory. Recall is the process of retrieving information from memory.

2019, Tulving et al., 1972]. Semantic memory refers to general knowledge memories acquired through conceptual and factual knowledge about the world. In the context of the open-world game, it can be considered the part that encapsulates the game rules and memories consistent with the relevant worldview. We divided semantic memory into two parts in the system. One is implemented with an external database, as its content is not frequently changed. Simultaneously, some semantic memories are stored in the long-term memory module in symbolic language.

Episodic memory refers to the memory of specific events that an individual experiences. These can be memories related to other players or Agents. In our memory system, we adopted a vector database in the long-term memory module for storing and retrieving these memories. Associated decay parameters are introduced as memories can be forgotten, with relevance scores decreasing over time. When reasoning with the LLM, such memory contents can be easily retrieved through vector queries.

Procedural memory refers to actions or skills that can be performed without conscious thought[Roediger, 1990], such as swimming, cycling, etc. These skills, with action properties, are represented as APIs in action space in our system. The action space is divided into public APIs and personal APIs. The personal APIs could be extended through learning [Sumers et al., 2023] which is mentioned in the section 4.

In the long-term memory module, we store all perceived memories in the semantic and episodic memory zones respectively. We propose a method named **Question-based Query**, which generates self-ask questions as queries that can be leveraged in search by vector similarity and predicate logic. This method facilitates the retrieval of semantic and episodic memories in the recall module, thereby improving the overall efficiency of memory utilization.

3.2 Working Memory

Working memory mainly holds the observational information and retrieved long-term memories needed for performing complex cognitive tasks (such as reasoning and learning) and interactive tasks [Baddeley, 2003, Miller et al., 2017]. These pieces of information are typically obtained through agents' observation as natural language data provided by the game side. Short-term memory, as its name suggests, represents a stage of memory that retains information for brief periods of time, generally only lasting a few seconds to a minute [Atkinson and Shiffrin, 1968]. To humans, the average capacity for retaining items in short-term memory is about 7±2, with a retention duration of roughly 20 to 30 seconds [Miller, 1956]. In this work, the two concepts are implemented as the same module, collectively referred to as working memory. In our architecture, it exists as a data cache from which information is extracted and dropped into the prompt's context. Its extraction process is further explained in more detail in the memory processing and decision-making sections.

3.3 Memory Processing

The memory processing module primarily handles the processing of memories that have been stored and are about to be stored. The three main stages of memory are encoding, storage, and recall [Melton, 1963]. Specifically, perceived input information is encoded and transformed into content in long-term memory, enabling it to be recalled in the space of long-term memory. In LARP, we simulate this procedure by processing all structured observational information provided in the game, combining it with retrieved content, and storing it in the working memory. This information serves as an input for a series of logic processing units in the decision-making module, continuously updating the content in the working memory. Once the length of working memory reaches a certain threshold, reflection is triggered, during which ineffective memories are filtered out, and the processed natural language memories and symbolic language memories are separately stored as episodic memory and semantic memory.

The core of memory encoding is the language transformation system. By aligning language models and probabilistic models, natural language is converted into a probabilistic programming language (PPL) [Wong et al., 2023] and logic programming language. PPL primarily handles probabilistic reasoning, while logic programming language pertains mainly to fact reasoning. Moreover, memory encoding should also be influenced by the consistency of prior knowledge, meaning that past knowledge will affect the current understanding [Bartlett, 1995]. The storage of memory has already been elaborated in the long-term memory section.

To humans, recall refers to the psychological process of retrieving information from the past. While in our architecture, it is the process of retrieving information from long-term memory. It first involves compound retrieval from long-term memory, including search by vector similarity and predicate logic. First, we employed self-ask strategies to form the queries, prompting LLM to raise questions regarding agents' observations, personalities, and experiences. After obtaining the queries, we adopted 3 methods to perform the retrieval. For logic programming search, LLM generates a query in a logic programming language that answers the self-ask questions based on available rules and facts. For similarity search, two methods were available. One method is using self-ask questions as queries for vector similarity search, matching with question-answer pairs in the vector database of episodic memory. The other method is using keywords extracted from the self-ask questions to match with the natural language memories in the same database. This process will be repeated until a final answer is obtained, and this can also be considered semantic retrieval [Press et al., 2022]. Figure 3 shows the detailed control flow.

Based on the recall capabilities, our architecture adopts CoT [Wei et al., 2022] to reason about the retrieved content and observed information and perform memory reconstruction, ie., using prior knowledge to influence observed facts to some extent [Loftus and Palmer, 1974] though the reconstructed memories might be distorted. In addition, we also simulated the process of human forgetting in the recall workflow. When the retrieval system operates, we introduce a decay parameter σ represented by Wickelgren's power law to mark the forgetting probability of this memory [Wixted and Carpenter, 2007]. The calculation formula is as follows:

$$\sigma = \alpha\lambda N(1 + \beta t)^{-\psi} \tag{1}$$

Here, λ represents the importance level, given by a scoring model. N stands for the number of retrievals of this memory, and t is the elapsed time after the last retrieval. ψ is the rate of forgetting for each character. α and β are

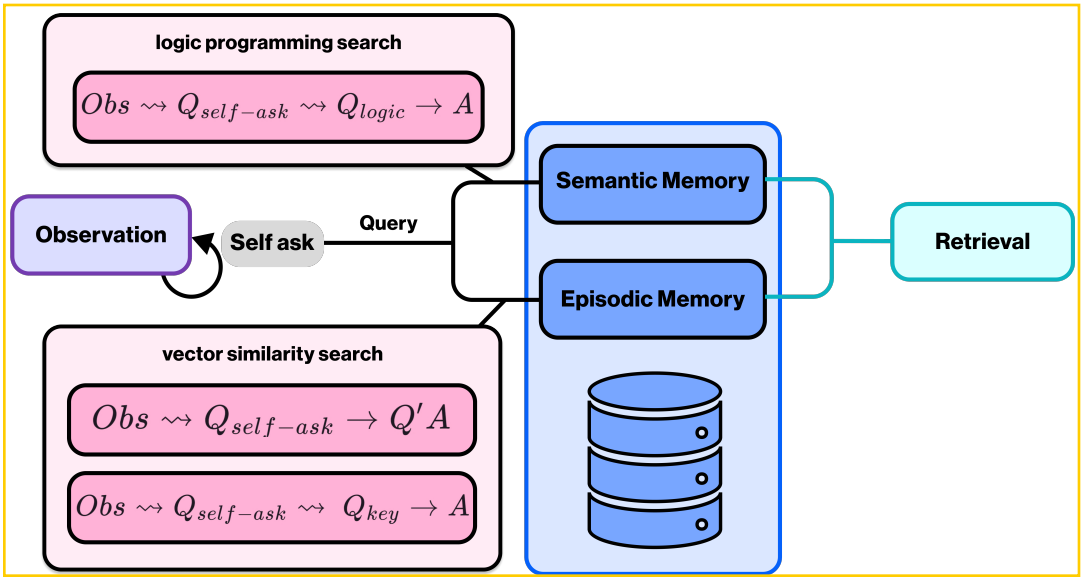

Figure 3: Detail control flow of recall psychological process. First conduct self-asking about the observation to get self-ask questions. Using the self-ask questions as queries, different methods of retrieval are undertaken. 1. Generate predicate logic statements in logic programming language and probabilistic programming language based on queries. 2. Conducting a vector similarity search after extracting keywords from the queries. 3. Searching for question-answer pairs based on sentence similarity between queries and the questions of question-answer pairs. $Q_{self-ask}$ means the self-ask questions which were used as queries, Q_{logic} stands for predicate logic query statements, Q_{key} is the extracted keywords, $Q'A$ stands for the question-answer pairs.

the scaling parameters for importance and time, respectively. Through multiple rounds of memory reconstruction and forgetting processes, our cognitive architecture can ultimately simulate instances of memory distortion.

3.4 Decision Making

The decision-making module produces final decisions under the joint effect of observation and working memory. The core section of the decision-making module is an ordered cluster of programmable units. Each unit will process the content in working memory and context, updating the results to the working memory in real time. These units can be simple information processing units, such as those conducting affective computing, or complex units equipped with specifically fine-tuned LLM models, like intent analysis and output formatting. These units are infinitely scalable and can handle all types of memory processing tasks. As each unit communicates with working memory, it updates the working memory in real-time, allowing the agents to react timely when observation changes during the process. The execution order of these units would be determined by a language model assistant. The final output of the decision-making module could be tasks or dialogue contents for the Non-Player Characters (NPCs).

4 Environment Interaction

For role-playing language agents in open-world games, generating tasks based on current observations through the cognitive architecture only accomplishes the objective within agents. However, in open-world games with free actions and rich gaming content, agents need to interact with the game environment by connecting the internal and the external. There are various works in employing language agents to interact with open-world game environments [Wang et al., 2023d, Zhu et al., 2023, Yang et al., 2023, Wang et al., 2023e]. For instance, Voyager uses the concept of an *automatic curriculum*, obtaining objectives by feeding GPT4 the contents and states of environmental observations. Then, GPT4 is prompted to generate the functioning code to reach the objectives. The paper also presents the *skill library* method, which embeds the description of the generated code as a key and the code as a value, achieving high

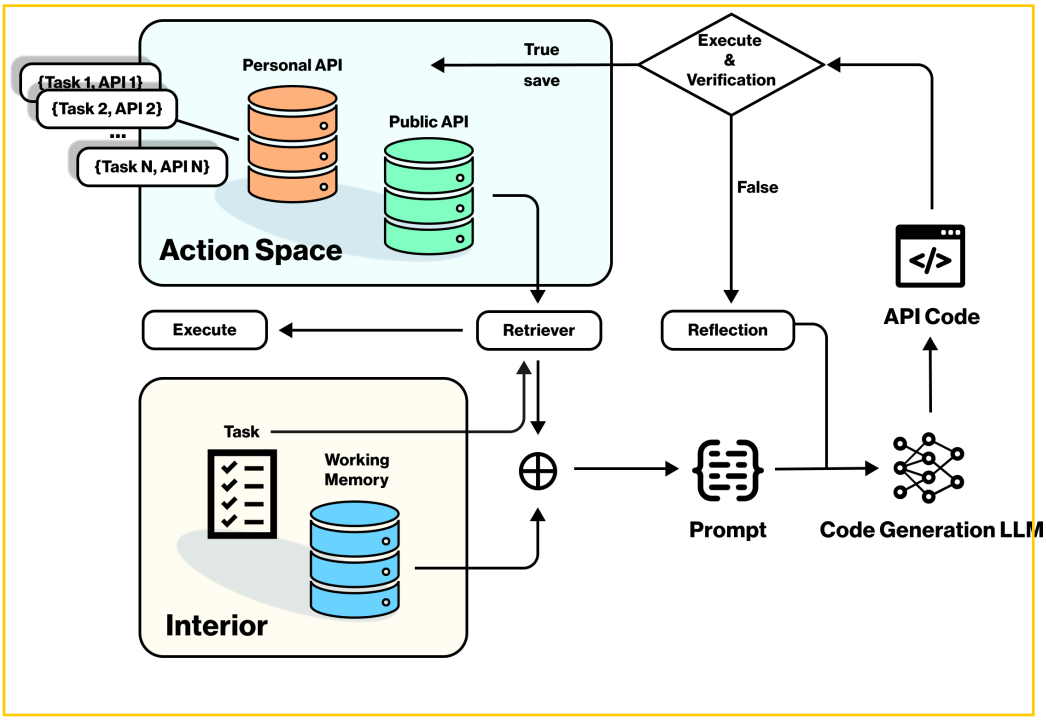

Figure 4: Environment Interaction.

extensibility of incorporating new skills by adding key-value pairs. Octopus utilizes the Visual Language Model (VLM) to acquire observations. However, this approach can lead to high dimensions of data feature distribution, resulting in poor controllability. Moreover, the operational cost of the VLM is high, and it's challenging to collect the data set's prior knowledge within the game.

Figure 4 shows the fundamental interaction procedural. Interior refers to the working memory and the tasks that need to be executed based on the current situation, generated by the observation and cognitive architecture. The Action Space is the agent's executable action APIs in the game world, encompassing both public and personal APIs. The personal API library stores tasks-API pairs, while the public APIs are basic actions. A personal API can be a series of basic actions, facilitating quick decision-making and reuse of APIs.

Once we have generated the corresponding plans in the decision-making module, we initially attempt to break down the overall task goal into several subtask goals. These subtask goals present as strictly ordered sequence-sensitive arrangements. For each task goal or subtask goal, the whole system will integrate it with the working memory. Then, it will use a retriever to search separately in the personal API library and public API library. If the action corresponding to the task already exists in the personal API library, the action is instantly performed. Otherwise, the system completes the corresponding prompt with the entire action space and interior content to generate the structured code using a fine-tuned LLM. Upon the successful execution and verification of the generated code blocks, they are stored as a new interface in the personal API library in the form of (Task, API) for future use. If verification fails, the reflection unit is activated to generate new code blocks [Shinn et al., 2023].

Simultaneously, we also collect the paired prompt and generated code as a training set for fine-tuning code generation LLM [Patil et al., 2023]. After the successful execution and verification, the results are fed back through RLHF to enhance the model's capabilities.

5 Personalities

The role of distinct personalities is vital in enhancing the cognitive capabilities of language agents in role-playing. Aligning variable personalities permits language models to better comprehend different perspectives and portray various cultures and social groups. Language models acting out different roles in complex scenarios must deeply understand, respond, and express in their unique ways. This necessitates the models to possess human-like thought processes with a wide range of personalities. Understanding the generation of diverse language expressions, handling multicultural content, and exhibiting varied thoughts, perspectives, emotions, and attitudes - all demand the model to accommodate these distinct personalities. Therefore, this section will delve into its implementation.

LARP adopts the strategy of simulating a cluster of models fine-tuned with various alignments to tackle the agents' diversified viewpoints. These models might be applied in different modules. During the training phase, we pre-trained several base models of different scales. Our pretraining datasets contain perspectives of different cultures and groups. After pre-training, these base models undergo supervised fine-tuning (SFT) on a series of instruction datasets of persona and character to enhance instruction-following and role-playing capabilities [Chen et al., 2023b, Dong et al., 2023]. This instruction dataset was established through data distillation based on question-answer pairs generated by SOTA models. Then, the dataset was optimized via assessment, modification, and adjustment based on human feedback.

It is possible to create multiple datasets and fine-tune LoRAs (Low-Rank Adaption) for capabilities such as reflection, code generation, and intent analysis. These LoRAs can be dynamically integrated with base models of different scales, creating a model cluster with diverse capabilities and personalities. These capabilities cover tasks such as language style, emotion, action generation, reflection, memory reconstruction, and more.

However, one of the main challenges in fine-tuning language models to construct different LoRAs for role-playing is acquiring quality data. Successful fine-tuning requires custom datasets of high quality, which need to be carefully constructed to capture various aspects of characters, including their language style, behavior style, personality traits, idioms, backstories, and more. The construction of datasets requires extensive literary creativity, script compilation, and character research to ensure that the generated language not only fits the character's persona and features but also interacts with the user in an appropriate manner [Wang et al., 2023b].

To enrich the diversity of the agent, we set up several post-processing modules, including the action verification module and the conflict identification module. The action verification module is part of the environment interaction module, which checks whether the generated actions can be correctly executed in the game. Conversely, within the cognitive architecture, the conflict identification module checks whether the decisions and conversations encompass conflicts with character relationships, personalities, and game worldview. When such conflicts are detected, the module will undertake actions like rejecting the result or rewriting it to prevent the agent from going out of character.

6 Discussions

6.1 Multi-Agent Cooperation and Agent Socialization

A single Language Agent role-playing under the framework proposed in this work is insufficient to solve the issue of creating rich content in open-world games. To bring each character supported by an Agent to life, a robust social network needs to be established. One possible approach is to build suitable sociological mechanisms and behaviors atop the large language model-driven agents to ensure that NPCs can still maintain their rationality and logic after extensive role-playing reasoning.

6.2 Confidence of Model Clusters vs. Evaluation and Feedback System

Combining language models and cognitive science makes language agents align more closely with genuine human cognition. This method effectively mitigates the problem of a single large model's inability to enhance role-playing outcomes due to insufficient data. Simultaneously, since the cognitive system only consists of domain tasks, fine-tuned small-scale models can achieve satisfactory performance. It saves costs compared to fine-tuning large models. However, due to the randomness of language model output results, it's unpredictable how the cumulative distortion of the results produced by each task affects the distortion of the entire cognitive architecture. It's hard to

say such a distorted agent could be called a human-believable agent. Therefore, a corresponding evaluation and a measurement framework are needed to impose constraints and convergence on the distortion of the cognitive system. Establishing a measurement and feedback mechanism for the entire system to measure the logical deviation of each logic unit can optimize system robustness and minimize the impact of single-system distortion on the overall system.

7 Conclusion

In this study, we present a language agent framework-oriented open-world games and elaborate on this framework from three aspects: cognitive architecture, environmental interaction, and alignment with diverse value perspectives. Addressing cognitive architecture, we employ more intricate techniques from cognitive science to enable the agent to make more reasonable decisions, alongside implementing post-processing constraints to prevent undue freedom in the agent, thereby bringing them closer to real human behavior in role-playing contexts. We envisage that our work harbors tremendous potential within the open-world games, breathing new life into this traditional domain, and ultimately catering the experience akin to that of 'Westworld'.

References

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

OpenAI. Gpt-4 technical report, 2023.

Stuart J Russell and Peter Norvig. *Artificial intelligence a modern approach*. London, 2010.

Michael Wooldridge and Nicholas R Jennings. Intelligent agents: Theory and practice. *The knowledge engineering review*, 10(2):115–152, 1995.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*, 2023a.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *arXiv preprint arXiv:2309.07864*, 2023.

Theodore R Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*, 2023.

Sahith Dambekodi, Spencer Frazier, Prithviraj Ammanabrolu, and Mark O Riedl. Playing text-based games with common sense. *arXiv preprint arXiv:2012.02757*, 2020.

Ishika Singh, Gargi Singh, and Ashutosh Modi. Pre-trained language models as prior knowledge for playing text-based games. *arXiv preprint arXiv:2107.08408*, 2021.

Shunyu Yao, Karthik Narasimhan, and Matthew Hausknecht. Reading and acting while blindfolded: The need for semantics in text game agents. *arXiv preprint arXiv:2103.13552*, 2021.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. Learning to speak and act in a fantasy text adventure game. *arXiv preprint arXiv:1903.03094*, 2019.

OpenAI, :, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d. O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning, 2019.

Kai Arulkumaran, Antoine Cully, and Julian Togelius. Alphastar: An evolutionary computation perspective. In *Proceedings of the genetic and evolutionary computation conference companion*, pages 314–315, 2019.

Jiaju Lin, Haoran Zhao, Aochi Zhang, Yiting Wu, Huqiuyue Ping, and Qin Chen. Agentsims: An open-source sandbox for large language model evaluation. *arXiv preprint arXiv:2308.04026*, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22, 2023.

Zekun Moore Wang, Zhongyuan Peng, Haoran Que, Jiaheng Liu, Wangchunshu Zhou, Yuhan Wu, Hongcheng Guo, Ruitong Gan, Zehao Ni, Man Zhang, et al. Rolellm: Benchmarking, eliciting, and enhancing role-playing abilities of large language models. *arXiv preprint arXiv:2310.00746*, 2023b.

Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. Character-llm: A trainable agent for role-playing. *arXiv preprint arXiv:2310.10158*, 2023.

Murray Shanahan, Kyle McDonell, and Laria Reynolds. Role play with large language models. *Nature*, pages 1–6, 2023.

Cheng Li, Ziang Leng, Chenxi Yan, Junyi Shen, Hao Wang, Weishi MI, Yaying Fei, Xiaoyang Feng, Song Yan, HaoSheng Wang, et al. Chatharuhi: Reviving anime character in reality via large language model. *arXiv preprint arXiv:2308.09597*, 2023a.

Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*, 2023b.

Dake Chen, Hanbin Wang, Yunhao Huo, Yuzhao Li, and Haoyang Zhang. Gamegpt: Multi-agent collaborative framework for game development. *arXiv preprint arXiv:2310.08067*, 2023a.

Chen Qian, Xin Cong, Cheng Yang, Weize Chen, Yusheng Su, Juyuan Xu, Zhiyuan Liu, and Maosong Sun. Communicative agents for software development. *arXiv preprint arXiv:2307.07924*, 2023.

Yuan Li, Yixuan Zhang, and Lichao Sun. Metaagents: Simulating interactions of human behaviors for llm-based task-oriented coordination via collaborative generative agents. *arXiv preprint arXiv:2310.06500*, 2023c.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155*, 2023.

Lei Wang, Jingsen Zhang, Xu Chen, Yankai Lin, Ruihua Song, Wayne Xin Zhao, and Ji-Rong Wen. Recagent: A novel simulation paradigm for recommender systems. *arXiv preprint arXiv:2306.02552*, 2023c.

Ruibo Liu, Ruixin Yang, Chenyan Jia, Ge Zhang, Denny Zhou, Andrew M Dai, Diyi Yang, and Soroush Vosoughi. Training socially aligned language models in simulated human society. *arXiv preprint arXiv:2305.16960*, 2023a.

Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Zhaopeng Tu, and Shuming Shi. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118*, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Chi-Min Chan, Weize Chen, Yusheng Su, Jianxuan Yu, Wei Xue, Shanghang Zhang, Jie Fu, and Zhiyuan Liu. Chateval: Towards better llm-based evaluators through multi-agent debate. *arXiv preprint arXiv:2308.07201*, 2023.

Marc-Alexandre Côté, Akos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Computer Games: 7th Workshop, CGW 2018, Held in Conjunction with the 27th International Conference on Artificial Intelligence, IJCAI 2018, Stockholm, Sweden, July 13, 2018, Revised Selected Papers 7*, pages 41–75. Springer, 2019.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7903–7910, 2020.

Zihao Wang, Shaofei Cai, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023d.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023e.

Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, et al. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory. *arXiv preprint arXiv:2305.17144*, 2023.

Lilian Weng. Llm-powered autonomous agents. *lilianweng.github.io*, Jun 2023. URL `https://lilianweng.github.io/posts/2023-06-23-agent/`.

Chenxu Hu, Jie Fu, Chenzhuang Du, Simian Luo, Junbo Zhao, and Hang Zhao. Chatdb: Augmenting llms with databases as their symbolic memory. *arXiv preprint arXiv:2306.03901*, 2023.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*, 2023.

Ziheng Huang, Sebastian Gutierrez, Hemanth Kamana, and Stephen MacNeil. Memory sandbox: Transparent and interactive memory management for conversational agents. In *Adjunct Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–3, 2023.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Bo Liu, Yuqian Jiang, Xiaohan Zhang, Qiang Liu, Shiqi Zhang, Joydeep Biswas, and Peter Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023b.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Hao Liu, Carmelo Sferrazza, and Pieter Abbeel. Chain of hindsight aligns language models with feedback. *arXiv preprint arXiv:2302.02676*, 3, 2023c.

Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. *arXiv preprint arXiv:2305.04091*, 2023f.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*, 2023.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

John E Laird. *The Soar cognitive architecture*. MIT press, 2019.

Endel Tulving et al. Episodic and semantic memory. *Organization of memory*, 1(381-403):1, 1972.

Henry L Roediger. Implicit memory: Retention without remembering. *American psychologist*, 45(9):1043, 1990.

Alan Baddeley. Working memory: looking back and looking forward. *Nature reviews neuroscience*, 4(10):829–839, 2003.

George A Miller, Galanter Eugene, and Karl H Pribram. Plans and the structure of behaviour. In *Systems Research for Behavioral Science*, pages 369–382. Routledge, 2017.

Richard C Atkinson and Richard M Shiffrin. Human memory: A proposed system and its control processes. In *Psychology of learning and motivation*, volume 2, pages 89–195. Elsevier, 1968.

George A Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

Arthur W Melton. Implications of short-term memory for a general theory of memory. *Journal of verbal Learning and verbal Behavior*, 2(1):1–21, 1963.

Lionel Wong, Gabriel Grand, Alexander K Lew, Noah D Goodman, Vikash K Mansinghka, Jacob Andreas, and Joshua B Tenenbaum. From word models to world models: Translating from natural language to the probabilistic language of thought. *arXiv preprint arXiv:2306.12672*, 2023.

Frederic Charles Bartlett. *Remembering: A study in experimental and social psychology*. Cambridge university press, 1995.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*, 2022.

Elizabeth F Loftus and John C Palmer. Reconstruction of automobile destruction: An example of the interaction between language and memory. *Journal of verbal learning and verbal behavior*, 13(5):585–589, 1974.

John T Wixted and Shana K Carpenter. The wickelgren power law and the ebbinghaus savings function. *Psychological Science*, 18(2):133–134, 2007.

Jingkang Yang, Yuhao Dong, Shuai Liu, Bo Li, Ziyue Wang, Chencheng Jiang, Haoran Tan, Jiamu Kang, Yuanhan Zhang, Kaiyang Zhou, et al. Octopus: Embodied vision-language programmer from environmental feedback. *arXiv preprint arXiv:2310.08588*, 2023.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *arXiv preprint arXiv:2305.15334*, 2023.

Hao Chen, Yiming Zhang, Qi Zhang, Hantao Yang, Xiaomeng Hu, Xuetao Ma, Yifan Yanggong, and Junbo Zhao. Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning. *arXiv preprint arXiv:2305.09246*, 2023b.

Guanting Dong, Hongyi Yuan, Keming Lu, Chengpeng Li, Mingfeng Xue, Dayiheng Liu, Wei Wang, Zheng Yuan, Chang Zhou, and Jingren Zhou. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*, 2023.