# The Game Boy Learning Environment

Edoardo Fazzari, *Member, IEEE,* Donato Romano, Fabrizio Falchi, and Cesare Stefanini, *Senior Member, IEEE*

***Abstract*—In this article, we introduce the Game Boy Learning Environment (GLE), an innovative suite based on Nintendo Game Boy games, crafted to advance and evaluate deep reinforcement learning algorithms on rich and varied gameplay tasks. GLE offers a comprehensive selection of eleven Game Boy environments, spanning nine distinct titles. These environments represent a significant leap in complexity compared to previous endeavors, like the Arcade Learning Environment, presenting challenges for reinforcement learning like intricate long-term planning, strategic foresight, and hierarchical decision-making, posing substantial difficulties even for proficient human players. We delineate the spectrum of available environments and furnish initial baseline results obtained through the development and assessment of intelligent agents, employing established AI methodologies to address individual levels or subtasks within these environments. All software is openly accessible to the public via https://github.com/edofazza/GameBoyLearningEnvironment.**

***Index Terms*—Deep Reinforcement Learning, Game Boy, Learning Environments, Artificial Intelligence**

## I. Introduction

In recent years, the landscape of artificial intelligence has witnessed significant advancements, particularly in Natural Language Processing (NLP), driven by the emergence of Large Language Models (LLMs) [6]–[8]. These models are capable of generating text and engaging in conversations with humans, exhibiting behavioral and personality traits that are statistically indistinguishable from those of random human beings [4]. Similarly, in the domain of image generation [9], AI has achieved capabilities comparable to those of professional artists and editors [10], [11]. However, despite this trend towards imbuing AI with human-like characteristics [12], we are still far from achieving artificial general intelligence [13], [25].

Presently, while AI models are proficient at tasks such as writing computer programs [14], composing music [15], writing stories [62], and playing various games, including chess [16], Go [17], board games [18], [19], and video games [20], [21], they lack the ability to plan over long time horizons and adapt their behavior dynamically. These are skills exhibited by human children—for instance, consider a child playing a video game. The child can autonomously discern the objective of the game without external prompting, and adjust their actions in real-time based on the information conveyed through words and visual cues on the screen. Furthermore, a child can recall actions performed hours earlier during gameplay; if they require a specific tool that was left somewhere hours before, they know how to retrieve it—a task that poses a significant challenge for current deep reinforcement learning agents. While these AI agents may outperform humans in many complex tasks, they often struggle with tasks that humans consider trivial, as noted by [5].

In this work, we introduce the Game Boy Learning Environment (GLE), a novel challenge platform designed for evaluating deep learning agents through the medium of video games. GLE is a Python framework, released under the MIT Licence, that interfaces with emulated Game Boy environments. The Game Boy, a fourth-generation game console released in 1989, boasts a rich library of games spanning diverse genres such as RPGs, adventures, puzzles, and action games (a subset of them are shown in Figure 1). Many iconic titles, such as Pokémon, were pioneered on this console. While modern game consoles offer semi-realistic visuals, advanced controls, and increasing complexity akin to real-world scenarios, Game Boy games occupy a compelling middle ground: they are significantly more complex and cognitively demanding than Atari 2600 titles [3], which are often treated as toy environments lacking realistic structure and constraints [22], yet remain far more lightweight and accessible than today's AAA games, which also currently lack standardized evaluation frameworks for reinforcement learning.

Games on the Game Boy frequently involve hierarchical decision-making and reasoning. Models trained in such environments have the potential to learn hierarchical representations of game states, offering insights into the application of hierarchical learning across various domains. Notable challenges encountered in Game Boy games include:

1) *Imperfect information*: Games typically provide only partial visibility of the map through a local camera that actively tracks the player.
2) *Diverse action space*: The same action can yield vastly different outcomes depending on the game context. For instance, pressing the `A` button may prompt the player to confirm a dialogue choice in a menu, execute an attack during combat, or interact with objects (e.g., opening doors or picking up items) while exploring the game world.
3) *Long-term consequences*: Games often span thousands or even millions of frames and actions. Early decisions made by the player may have far-reaching consequences that may not manifest until much later in the game or in distant locations from where the initial decision was made.

In summary, our contributions are as follows:

All authors are with the BioRobotic Institute, Sant'Anna School of Advanced Studies, Pontendera, 56025, Italy, (e-mail: edoardo.fazzari@santannapisa.it, donato.romano@santannapisa.it, cesare.stefanini@santannapisa.it).

Edoardo Fazzari, Donato Romano and Cesare Stefanini are with the Department of Excellence in Robotics and AI, Sant'Anna School of Advanced Studies, Pisa, 56127, Italy.

Edoardo Fazzari and Fabrizio Falchi are with the Institute of Information Science and Technologies, National Research Council of Italy, Pisa, 56124, Italy (e-mail: fabrizio.falchi@cnr.it).

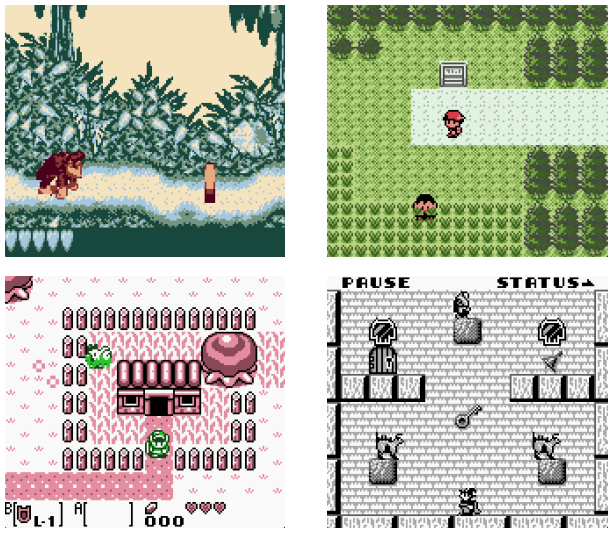


Fig. 1. Some of the environments available in GLE: Donkey Kong Land (top-left), Pokemon Silver (top-right), The Legend of Zelda: Link's Awakening (bottom-left), Solomon's Club (bottom-right).

- Development of the Game Boy Learning Environment (GLE), our primary contribution, designed to train and evaluate deep reinforcement learning agents in more complex game environments than previously explored.
- Definition of desired goal (which we called ultimate task) and subtasks for each game, providing possible benchmarks for agent performance evaluation and enabling multi-task and curriculum learning.
- Demonstration of GLE's effectiveness through experiments using Proximal Policy Optimization (PPO) [24], Deep Q Network (DQN) [57], Soft Actor-Critic (SAC) [66] and Twin Delayed DDPG (TD3) [67] to address various game levels and subtasks.
- Development of a software tool for collecting human experience by playing games.

The introduction of new domains, such as Game Boy games, has the potential to advance deep learning and reinforcement learning research, much like previous introductions of novel environments [3], [37], [44]. Therefore, ensuring the availability of challenging environments beyond current model capabilities is crucial.

## II. Related Work

Employing games in Reinforcement Learning (RL) finds its roots in 1995 when Temporal Difference Learning was applied to play Backgammon [26]. Subsequently, numerous other games were explored [27]–[29], garnering significant attention within the research community and culminating in the establishment of the *General Game Playing* (GGP) and the *General Video Game AI Competition* (GVGAI) [30]–[32]. While GGP ran from 2005 to 2016, and GVGAI from 2014 to 2021 (with its learning track introduced in 2017), the field experienced a significant surge in 2013 following the introduction of Atari games [3] and Deep Q-Learning [33], which combined RL with Deep Learning. This breakthrough not only revolutionized RL in gaming but also extended its applications to robotics [34]. However, even if certain Atari games, such as Montezuma's Revenge [35], [60], posed formidable challenges, the need for more complex environments alongside the advent of Deep Q-Learning arose.

In 2016, strategic games like Shogi, chess, and Go regained prominence with the emergence of *AlphaGo* [17], subsequently evolving into *AlphaZero* [42] and *MuZero* [43]. Notably, *MuZero* demonstrated versatility by excelling not only in strategic games but also in playing Atari games. Concurrently, *Project Malmo* [37], a research platform built on Minecraft, was introduced, offering a complex virtual environment ideal for studying and advancing RL techniques [38]. This led to the launch of a competition based on the MineRL dataset in 2019 [39], [40], initially relying on human demonstrations but later transitioning to autonomous task completion [41].

In 2017, Blizzard introduced the StarCraft II Learning Environment (SC2LE) [44], providing a multi-agent problem setting where collaboration is essential. Similar to developments in Go and chess, AlphaStar surpassed human performance, achieving Grandmaster level [21]. Subsequently, RL algorithms were applied to other modern video games such as Dota 2 [46], Gran Turismo [45], and Counter-Strike Deathmatch [47], expanding the scope beyond old shooting games like Quake III [48]. Recently, the SIMA project introduced asynchronous environments based on recent computer video games, enabling agents to perform short-horizon tasks guided by user-generated language instructions[1].

While RL has predominantly focused on visual games, text-based games serve as valuable benchmarks for language-based agents [49]. These games offer textual observations and actions, which can either be pre-generated [51] or generated dynamically using verb-object combinations [50]. Moreover, language plays a crucial role in games involving cooperation and competition, as seen in Diplomacy, where RL algorithms like Cicero learn to generate dialogue to advance strategic plans [52].

Recent research endeavors have shifted towards developing foundation agents capable of dynamic adaptation across a range of applications [53]. CRADLE [54], for instance, demonstrates this transition, being an agent proficient in navigating the intricate storyline and missions of AAA games like Red Dead Redemption II.

For Game Boy games specifically, research is scarce. The only existing studies are those focusing on Tetris, employing a genetic algorithm agent [64], and utilizing Monte Carlo approaches for Pokemon battles [63]. However, a recent YouTube video titled "Training AI to Play Pokemon with Reinforcement Learning"[2] showcases the training of an agent on Pokemon Red, successfully achieving the second gym badge.

### A. How GLE Differs

Game Boy titles encompass a broader range of genres and gameplay mechanics compared to the more arcade-like Atari games [3]. Game Boy games include role-playing games

[1]https://deepmind.google/discover/blog/sima-generalist-ai-agent-for-3d-virtual-environments/

[2]https://github.com/PWhiddy/PokemonRedExperiments

(RPGs), platformers, puzzle games, and adventure games, each with its own set of rules, challenges, and design elements. Many Game Boy games feature more complex level designs, narratives, and player progression systems than typical Atari games. For example, the inclusion of RPG elements, inventory management, and multi-layered puzzles in Game Boy titles can present a more nuanced challenge for RL agents. These games often have non-linear gameplay and exploration, as seen in titles like The Legend of Zelda: Link's Awakening or Final Fantasy Adventure. In these games, agents must navigate complex game worlds, solve intricate puzzles, and backtrack through levels, requiring skills like exploration, planning, and decision-making. In contrast, Atari games tend to be more linear, with a clear progression of difficulty but less emphasis on exploration and long-term planning. Atari games usually feature simpler, often repetitive gameplay loops focusing on score maximization.

Furthermore, RPGs require players to comprehend textual information presented throughout the game in order to progress effectively. This includes dialogues between playable and non-playable characters, environmental cues such as signs, and menu management tasks like selecting actions (e.g., the names of Pokémon attacks). For instance, in Pokémon Red, the player learns how to proceed by interacting with various characters at the start of the game. The player's mother in their house directs the player to the professor's lab, and the professor's assistant then instructs the player to go to the grass to meet the professor. Successful navigation of these initial narrative elements is crucial, as some events may be locked out if certain tasks are not completed in the proper sequence, demonstrating the hierarchical decision-making required of the player. Thorough understanding of the textual information presented is, therefore, a vital skill for RPG players to master in order to advance through the game's storyline and overcome challenges. This represents a significant cognitive demand placed on the player throughout the RPG experience.

The game set we provide includes not only RPGs, but also platform and puzzle games. The platform games, such as Donkey Kong Land and Super Mario Land, are more linear than the previous RPG titles, but they often incorporate hidden areas, secret paths, and collectibles that encourage exploration. For a RL agent, discovering these secrets may require advanced exploration strategies, owing to the partial observability of the environments, which distinguishes these games from the full observability typically found in Atari games. The Game Boy platform games also introduce the concept of limited lives and power-ups (e.g., the Super Mushroom, Fire Flower in Super Mario Land, and animal buddies in Donkey Kong Land). Effectively managing these resources could be crucial for an RL agent's success, as it may help the agent learn about risk management and strategic planning. Additionally, the presence of checkpoints within levels could be leveraged to teach RL agents about sub-goals and long-term planning, as they need to strategize not only for immediate survival but also for progression toward these mid-level goals. These games also often use sound effects to signal important events, such as the acquisition of a power-up or the approach of an enemy. Incorporating audio cues into RL training could help develop more sophisticated agents that can use multimodal inputs for decision-making.

For puzzle games, we introduced Solomon's Club, which shares some gameplay mechanics with the Atari game Montezuma's Revenge. While both games involve solving challenges within a time limit, Solomon's Club emphasizes problem-solving abilities over reflexes and precision. In contrast, Montezuma's Revenge is often considered more complex for humans, as it demands precise jumping, timing, and navigation through a maze-like structure that can disorient players. Notably, Montezuma's Revenge levels contain all the necessary elements to complete the level within the same stage. In Solomon's Club, however, objects collected in earlier levels can be very helpful for progressing through later stages, facilitating completion and introducing an additional layer of strategy and planning. Furthermore, Solomon's Club's unique gameplay mechanic of creating and destroying blocks to build steps or access different parts of a level introduces further challenges for learning agents.

Another puzzle game included in GLE is Tetris. Although the game has been extensively studied and effectively solved using hand-crafted agents and genetic algorithms based on simplified and engineered feature representations of the game space [71], it remains an open challenge when approached using only raw RGB observations [72], making Tetris a valuable and relevant addition to the GLE suite.

The games included in our suite span three major categories—role-playing, platform, and puzzle games—each offering distinct challenges and opportunities that are particularly well-suited for developing and evaluating reinforcement learning agents. Below, we describe the key properties of each category and how they contribute to reinforcement learning research (see Table I for the classification of each environment):

- **RPGs.** As described in [74], RPGs feature movement across diverse maps that often differ significantly in layout and are not fully visible on screen at any one time. Some areas require alternative traversal methods, such as riding a vehicle to cross oceans, beyond simple walking. In addition to exploration, these games incorporate combat mechanics, either in real time or turn-based formats, increasing their complexity for autonomous agents. Furthermore, RPGs typically include puzzles, interactive dialogues with non-player characters that provide hints or guide progression, and extensive menu navigation for configuration, managing items and equipment. These multifaceted demands make RPGs particularly rich and challenging environments for reinforcement learning research.
- **Platform Games.** Platform games generally feature straightforward gameplay mechanics with a clear spatial objective, often moving from left to right to complete a level [75]. They present structured goals such as collecting items, avoiding hazards, defeating enemies, and reaching checkpoints, all offering partial observability of the environment like RPGs. Boss battles introduce more complex decision-making and combat dynamics, requiring agents to balance aggression with survival. Power-ups and temporary abilities add further layers of strategy,

encouraging agents to learn how to exploit environmental features and time their actions appropriately. Progress in these games often depends on executing specific action sequences, such as jumping over obstacles while dodging projectiles, supporting the development of precise motor control and long-horizon planning.

- **Puzzle Games.** Puzzle games on the Game Boy emphasize logical reasoning, spatial manipulation, and precise timing. These games typically require executing a sequence of actions in a specific order to solve problems [77]. Unlike more action-oriented genres, puzzle games focus on altering the game state through operations such as moving blocks, rotating elements, or activating switches within constraints like limited moves or time. Solving these puzzles often demands forward planning, pattern recognition, and adaptation to increasing complexity. Titles such as Tetris and Solomon's Club exemplify how these games foster decision-making under time pressure and delayed reward acquisition, core challenges in reinforcement learning. As such, they promote exploration of abstract problem-solving and generalization across varying level designs and rule sets [76].

*1) Comparison with DMLab-30:* DMLab-30 [23], developed by DeepMind, is another crucial environment for benchmarking in RL and serves as a valuable comparison point with our proposed GLE. DMLab-30 focuses on specific tasks such as 3D maze navigation, object location memory, and simple puzzle-solving. While this specialization allows for precise benchmarking of certain skills, it lacks the diversity and complexity inherent in role-playing games (RPGs), which often feature more varied and less linear tasks. DMLab-30's environments are tightly controlled and task-oriented (e.g., finding keys, collecting objects), but they do not offer the rich, multifaceted scenarios seen in Game Boy games, which include a broad array of objects, locations, characters, and interactions.

Additionally, DMLab-30 offers 3D environments, which contrasts with the 2D or top-down perspective of Game Boy games. However, the complexity and variety of gameplay scenarios in Game Boy titles are generally superior, involving a wider range of tasks and interactions, such as engaging with non-playable characters (NPCs), managing inventory, and navigating complex storylines. While DMLab-30 can integrate language tasks—such as question-answering and object selection—our environment could similarly incorporate text-based challenges by leveraging the extensive online availability of official Game Boy guides and manuals. These resources could aid in training agents, much like how they were previously used for Civilization II [65].

Moreover, unlike DMLab-30, many Game Boy titles—particularly RPGs in the GLE benchmark—require agents to comprehend textual instructions presented visually within the game world. These textual cues, such as dialogue from non-playable characters (NPCs) or menu-based prompts, are essential for determining how to progress in the game. Importantly, this information is not directly accessible through the system's RAM. While some reinforcement learning benchmarks allow access to emulator RAM for internal state inspection, in the case of GLE games, narrative instructions and quest information are not stored at fixed, interpretable memory addresses. For example, in The Legend of Zelda: Link's Awakening, a player must understand the owl's on-screen message instructing them to head north after acquiring a sword. This message is embedded in the visual rendering of the scene rather than stored in a clearly identifiable RAM location. As such, agents must exhibit a degree of visual language understanding, which introduces a unique challenge. This complexity makes GLE a compelling testbed for integrating large language models (LLMs) into agents for high-level reasoning and narrative-based decision-making.

*2) Comparison with NES:* Another relevant platform for comparison with GLE is the Nintendo Entertainment System (NES). However, unlike the Arcade Learning Environment (ALE), there is no unified or officially maintained reinforcement learning suite for NES games. Instead, NES-based research has focused primarily on specific titles or groups of titles, with notable attention given to Super Mario Bros., Mega Man, and Arkanoid [73]. Released before the Game Boy, the NES is a console designed to connect to a TV screen, which distinguishes it from the Game Boy. Additionally, many NES games, because of being older, share characteristics with those from the Atari console. Titles such as Arkanoid are simply evolutionary developments of Atari games like Breakout, offering limited complexity compared to later Game Boy titles. Although the NES introduced new games such as Final Fantasy and Dragon Warrior, the Game Boy features updated versions of these titles, as well as a wide range of subsequent releases, including those for the Game Boy Color, providing a better suite of available games. From a hardware perspective, NES games operate at a higher resolution (256×240 pixels) compared to the Game Boy's 160×144 display. This increased visual complexity results in a larger input space when training agents purely from pixels, possibly leading to higher computational costs and slower convergence during learning. In contrast, the Game Boy's lower resolution and simplified color palette reduce the dimensionality of the input space, while still retaining substantial semantic richness. For these reasons, we have built our learning environment around Game Boy games, as it strikes a favorable balance between game complexity and the simplicity of information representation, making it more suitable than NES for our purposes.

## III. Game Boy Learning Environment

In this section, we delineate our principal contribution, the Game Boy Learning Environment (GLE). GLE stands as a software crafted with the express purpose of providing a suite of 11 video games, configured as `gymnasium` environments [1], aimed at facilitating the development of intelligent agents capable of engaging in gameplay within these specified gaming domains.

### A. The Nintendo Game Boy

The Game Boy is Nintendo's first handheld game console released in 1989 and sold for over a decade. An estimated

118.69 million units of the Game Boy and its successor, the Game Boy Color, have been sold worldwide, making it the fourth best-selling video game console of all time [2]. Over 1000 games were released for the console, including titles as *Pokemon Red* and *Super Mario Land*, which became iconic symbols of this console.

In terms of hardware architecture, the Game Boy is characterized by its simplicity. Featuring a 4.19 MHz custom 8-bit CPU, it can be emulated at speeds exceeding real-time on modern computing hardware. The cartridge ROM, ranging from 4KB to a maximum of 1MB, serves as the repository for game code, while the console's internal RAM is limited to 8KB, with an expanded capacity of 32KB for the Game Boy Color iteration. Display capabilities include a resolution of 160 pixels width by 144 pixels height, supporting 2-bit color depth, while the Game Boy Color offers enhancements with a palette of 32k colors (56 simultaneously displayed). User interaction is facilitated through four operation buttons labeled `A`, `B`, `SELECT`, and `START`, alongside a directional pad.

### B. Interface

GLE builds on top of PyBoy[3], an open-source Python, based Game Boy emulator that provides low-level emulation capabilities such as screen rendering, RAM access, and input handling. While PyBoy serves as the core emulator, similar to how Stella[4] functions for Atari 2600 games, GLE plays a role akin to the Arcade Learning Environment (ALE): it wraps the emulator with an abstraction layer tailored for reinforcement learning. This layer standardizes diverse Game Boy games into learning environments by exposing structured observations, scores, and termination conditions. By default, each observation consists of a single game screen, represented as a 3D array of RGB images with dimensions 160 pixels wide by 144 pixels high. The action space comprises eight buttons defined by the joystick controller, with combinations thereof possible, allowing for advanced moves/actions in certain games. Moreover, the game-handling layer defines game-specific combination actions that enhance gameplay. Optionally, the `START` button can be toggled, as its default function may disrupt the agent's learning process by pausing or accessing menus. During real-time execution, the emulator generates 60 frames per second, capable of emulating up to 395 times real-time speed. Rewards are game-specific and can be based on score differentials or achieving specific tasks, with detailed descriptions provided in Section III-C1. Episode initialization and termination depend on the individual game; episodic environments start with the first frame of the first level, while continuous environments begin after the game introduction (further elucidated in Section III-C3). Eleven games are accessible through a unified interface, with the addition of support for new games being relatively straightforward. We selected 11 Game Boy games based on the online availability of well-documented RAM addresses, diversity of gameplay styles, and their popularity. These factors ensure the environments are technically accessible, semantically rich, and relevant in reinforcement learning. To expand this resource, in the *Supplementary Material*, we propose a submission process that enables the community to contribute detailed explanations of RAM entries for additional games.

Furthermore, GLE offers the capability to save and restore the emulator's state. Upon issuing a save-state command, GLE preserves all pertinent data, including RAM contents, registers, and address counters. Similarly, the restore-state command reverts the game to a previously saved state.

In addition to the environments, we developed a software tool integrated into GLE that allows humans to play the simulated game under the same conditions as the agent, including identical frame-skipping settings. This tool facilitates the collection of state and action data, as well as RAM information, during gameplay. We believe this tool represents a significant resource for enabling imitation learning within the GLE environments.

### C. GLE Environments Characterization

GLE comprises 11 distinct video games, as described in Table I, that were extremely popular on Game Boy. These games, spanning from puzzle (e.g., Solomon's Club) to RPG/adventure (e.g., The Legend of Zelda: Link's Awakening), endow GLE with a broad spectrum of environments, featuring both episodic and continuous gameplay experiences. Episodic environments encompass puzzle and platform games, where players typically navigate obstacles and enemies to progress towards completing levels, often culminating in boss encounters. On the other hand, continuous environments mirror the characteristics of "Artificial General Intelligence Environments (AGI), Task, and Agents" [25], [58].

While each environment varies slightly in minimum required actions, provided information (e.g., RAM data), and reward structure (explained in subsequent sections), they share a common observable definition—a RGB game frame of 160 width and 144 height. Specific details regarding minimum actions and RAM information for each environment are provided in the *Supplementary Material*, dedicated to the respective game of interest. Regarding termination criteria, two primary strategies are employed: termination upon loss of all lives or after a predefined number of action steps. We adopt the former strategy, in line with recommendations [55], as terminating upon loss of one life can adversely affect agent performance, despite its potential for teaching avoidance strategies [56]. Additionally, we employ a threshold-based termination approach, following existing literature [57], particularly crucial in continuous environments with end goals that are distant and challenging. Moreover, during experimentation, we introduce supplementary termination signals upon achieving specific subtasks.

*1) Rewards:* Even though our primary contribution is the release of GLE to support the research community in developing and evaluating agents in novel environments, we additionally propose reward definitions tailored to each environment based on available RAM information for experimental purposes. Generally, for puzzle and platform games, we assign

[3]https://github.com/Baekalfen/PyBoy?tab=readme-ov-file
[4]https://stella-emu.github.io

TABLE I
AVAILABLE GLE ENVIRONMENTS AND NUMBER OF SUBTASKS FOR EACH OF THEM. WE ALSO DEFINE THE ULTIMATE TASK FOR EACH GAME TO BE CONSIDERED COMPLETE, AND THE GAME CATEGORY. DEFEATING THE LEAGUE IN POKEMON MEANS BEATING THE FINAL SERIES OF TOUGH BATTLES, USUALLY THE ELITE FOUR AND THE CHAMPION, TO BECOME THE TOP TRAINER IN THE GAME.

| Game Title | Ultimate Task | Available Subtasks | Category |
|---|---|---|---|
| *Castlevania: The Adventure* | Complete the game | 1 | Platform |
| *Castlevania II: Belmont's Revenge* | Complete all levels | 11 | Platform |
| *Donkey Kong Land III* | Complete all levels | 5 | Platform |
| *Final Fantasy Adventure* | Complete the story | 3 | Role-playing |
| *Kirby's Dream Land* | Complete all levels | 8 | Platform |
| *Pokemon Red and Blue* | Defeat the League | 27 | Role-playing |
| *Pokemon Gold and Silver* | Defeat the League | 29 | Role-playing |
| *Super Mario Land* | Complete all levels | 12 | Platform |
| *Tetris* | Complete all levels | 10 | Puzzle |
| *The Legend of Zelda: Link's Awakening* | Complete the story | 4 | Role-playing |
| *Solomon's Club* | Complete all levels | 50 | Puzzle |

rewards based on the difference in score before and after taking an action. However, if the distance traversed by the character is available, we replace the score difference with the difference in distance traveled, as previously used in Super Mario Bros for NES [59]. Reward shaping for RPG/adventure games poses challenges and is limited to attempting to achieve specific subtasks within the environment, such as receiving a reward each time the boss's health decreases or when a subtask is completed. For more detailed information about the rewards employed, please refer to the *Supplementary Material*.

*2) Tasks:* In Table I, we define the ultimate task and number of available subtasks for each environment. For most environments, we have established multiple subtasks, detailed in the *Supplementary Material*, which open the possibility of training agents through curriculum learning [68] and multi-task learning [69].

The subtasks vary in complexity, ranging from short-term objectives, such as obtaining the shield in The Legend of Zelda: Link's Awakening (where the agent interacts with a character in the first game room), to longer-term goals, including reaching a specific character level in Final Fantasy Adventure or defeating the first gym in Pokémon games. In contrast, the ultimate task challenges the agent to complete the entire game, posing significant challenges for future research. In general, subtasks for puzzle and platform genres concentrate on single level completion or individual boss fights (e.g., in Kirby's Dream Land), whereas role-playing games offer subtasks involving exploration, item collection, and character statistic improvement.

The evaluation of certain subtasks is intended to demonstrate the ease with which our GLE can be used for agent training and deployment. To this end, we have also provided code examples to further facilitate the process.

*3) RAM States:* Each environment can load RAM data to restart the game from a saved position. This feature is particularly useful for two reasons: first, it enables the implementation of learning strategies that require starting actions from a non-initial position [60]; secondly, it allows skipping introductory cut-scenes, game instruction introductions, or starting the game at specific levels. For example, in our experiments, we tested whether an agent could defeat the first boss in Final Fantasy Adventure by loading the RAM information when the battle starts, thus bypassing all the introductory segments that involve naming characters. We provide multiple RAM states along with our code, and further details regarding each available RAM state are described in the *Supplementary Material*.

*4) Sound Information:* Although all tasks in our experiments are trained using visual (frame-based) observations, our framework also supports audio data access at each environment step. By setting the `return_sound` parameter to `True` during environment initialization, users can retrieve the corresponding sound buffer for each screen frame. The audio is returned as a stereo array of 801 samples across two channels. This feature facilitates research in multimodal deep reinforcement learning, where sound can serve as a complementary input modality, particularly valuable in games where audio cues convey critical information not visible on screen (e.g., an off-screen enemy preparing an attack or a player being hit) [70].

### D. Limitations

The main limitation of the Game Boy Learning Environment (GLE) lies in the limited availability of detailed documentation for RAM addresses in many Game Boy titles, which can hinder the extraction of specific game states needed for reward shaping. However, this is a common challenge in retro game research and is partially mitigated by existing community efforts, such as *Data Crystal*[5], and the availability of open-source assembly code for several games. Once the necessary memory addresses are identified, integrating new environments into GLE is straightforward. Our `GeneralGameBoyEnv` class provides all the functionality needed to interact with any Game Boy game, and users only need to overload the `step` function to define a game-specific reward and, optionally, the variables to track returned by the information parameter of the function. This makes the addition of new environments simple, even for less-documented games. Moreover, even in cases where RAM information is unavailable, such games can still serve as valuable benchmarks for evaluating generalization. For instance, an agent trained on Donkey Kong Land III can be tested on Donkey Kong Land, allowing for qualitative assessment of transfer learning and visual performance.

[5]https://datacrystal.tcrf.net/wiki/Data_Crystal

## IV. Experiments

In this section, we describe the experiments conducted to assess the usefulness of the environments provided by GLE by training agents to solve available subtasks. A complete report of our results using the GLE library is given in the *Supplementary Material*.

### A. Training details and policy details.

All agents were trained using frame pixel information processed by Recurrent PPO [24], DQN [57], SAC [66] and TD3 [67] with a convolutional neural network policy. The Convolutional Neural Network (CNN) architecture consisted of three convolutional layers [57]: the first had 32 filters of size 8x8 and a stride of 4, the second had 64 filters of size 4x4 and a stride of 2, and the third had 64 filters of size 3x3 and a stride of 1. This was followed by two fully connected layers, the first with 512 units. ReLU activation functions were used after each convolutional and fully connected layer, while the final output layer is feed to a Recurrent Neural Network (RNN) core that outputs the action. The training process involved a total of 500 thousand timesteps, with model updates occurring every 5120 steps for 5 epochs in case of PPO, and employing a buffer size of $5120 \times 5$ for the other models. In addition, PPO and DQN operate on discrete action spaces, whereas SAC and TD3 require continuous ones. To accommodate this, we adapted the action representations based on the algorithm used. Specifically, we implemented a `DiscreteToContinuousWrapper` that maps discrete actions to a continuous space by encoding each action as a one-hot-like vector with values bounded between –1 and 1. During execution, the continuous action output by the agent is converted back to a discrete action using an $\arg\max$ operation. This approach ensures compatibility with continuous control algorithms while preserving the semantic integrity of the original discrete action space.

We used the default hyperparameter settings provided by the `stable-baseline3` [61] library. Environment observations were normalized using the `NormalizeObservation` `Gymnasium` wrapper. All training was conducted on a single NVIDIA A100 40GB GPU and typically completed within a few hours.

### B. Evaluation

Our results obtained by training game-specific agents on the tasks described in Table I show that while some learning progress is already possible in Game Boy games, much more work remains to be done. Figure 2 shows a small subset of trained subtasks from 9 games.

Across the RPG cohort, PPO demonstrated the most consistent, albeit often modest, learning progress, particularly in tasks with more constrained state spaces or clearer intermediate goals. For instance, PPO agents exhibited discernible learning in Pokemon Red for the "Get Out House" subtask and in The Legend of Zelda: Link's Awakening for "Recover Shield". This relative efficacy of PPO might be attributed to its on-policy nature and its clipping mechanism, which can offer more stable policy updates and a balanced exploration-exploitation trade-off in certain scenarios [24]. However, for tasks demanding extensive navigation through larger environments or the completion of multi-stage quests (e.g., "Reach Boss" in Final Fantasy Adventure, "Reach Toronbo Shore" in Zelda, all tested algorithms, including PPO, generally failed to achieve significant positive rewards. This underscores the profound challenge of long-term credit assignment and efficient exploration in these sparse reward landscapes. SAC showed sporadic learning spikes in some long-horizon tasks, such as "Reach Cherrygrove City" in Pokemon Silver, potentially due to its entropy-driven exploration encouraging wider coverage of the state space [66], but this was not consistently superior. TD3, an algorithm primarily designed for continuous action spaces, consistently showed minimal to no learning, as anticipated in these discrete-action environments.

The difficulties observed align with broader research in applying RL to RPGs. For example, work on Pokemon Red by [78], highlighted the immense state space and exploration challenges. The struggle with sparse rewards and long-term planning is a well-documented issue in RL, often motivating research into hierarchical reinforcement learning [80], intrinsic motivation [79], and curriculum learning to guide agents through complex tasks. Our results suggest that standard RL architectures require such enhancements to make substantial progress in the more demanding Game Boy RPG scenarios. The partial successes in constrained tasks are valuable, but scaling to full game completion remains an open challenge.

Platform games, including Super Mario Land, Castlevania II: Belmont's Revenge, Castlevania: The Adventure, and Donkey Kong Land 3, necessitate precise motor control, reactive decision-making, and navigation through structured, often dynamic, levels. DQN frequently emerged as a strong performer for level completion tasks. This was particularly evident in Super Mario Land, where DQN agents consistently achieved high rewards across multiple levels, and in Donkey Kong Land 3 - Level 1, where DQN significantly outperformed other agents. This resonates with DQN's foundational success in discrete action environments, particularly those with clear goal states and reactive gameplay, akin to its achievements in the Atari Learning Environment [57]. Most of the time PPO and SAC also demonstrated competitive performance, even if their performance was not the best. However, the performance across different levels of Castlevania II: Belmont's Revenge was more varied, with PPO, DQN, and SAC each exhibiting superiority in different specific levels, suggesting that subtle differences in level design or enemy patterns can significantly influence algorithmic efficacy. One the other hand, the task of defeating bosses, which often requires different strategies than level traversal (e.g., pattern recognition, targeted attacks), saw SAC noted as effective. In the Final Fantasy Adventure - Boss Battle task, which features platformer-like combat mechanics, DQN and PPO showed strong initial learning, indicating their capability in such focused combat scenarios. TD3's performance was consistently poor, reinforcing its unsuitability for these discrete-action platformers.

The success of DQN in Game Boy platformers builds upon its legacy in Atari games. However, Game Boy platformers

often introduce increased complexity, such as more intricate level designs, multi-layered backgrounds (affecting perception if learning from pixels), more diverse enemy behaviors, and more nuanced physics compared to many Atari 2600 titles. The instances where PPO or SAC were competitive or superior could indicate that their policy gradient or actor-critic frameworks, perhaps with more explorative characteristics (especially SAC's entropy term), offer advantages when dealing with slightly more complex state-action dependencies or when a degree of stochasticity in policy is beneficial. Research in procedural content generation for platformers has also shown that RL agents can learn to generalize across varying levels, but often require robust feature representations and careful reward shaping [81].

Results were less promising when training agents on Solomon's Club, a game requiring players to remove specific blocks, retrieve keys, and enter doors. This game posed significant challenges, even though it was the only game with a fully observable game space, where the entire scene is visible to the player. No single algorithm consistently outperformed others; PPO, DQN, and SAC showed fleeting moments of learning or achieved moderate rewards on specific levels (e.g., SAC and DQN in 1-1, PPO with spikes in 1-2 and 1-6 before SAC's more stable rise), but mastery remained elusive. This difficulty highlights the game's significant combinatorial depth and the need for multi-step planning. The challenges in Solomon's Club are highly reminiscent of "hard-exploration" problems identified in other domains, most notably Montezuma's Revenge from the Atari suite [56]. Standard RL algorithms often struggle in such environments due to sparse rewards and the necessity of executing long, precise sequences of actions to achieve any positive feedback. The success of algorithms like GoExplore [36] on Montezuma's Revenge, which explicitly remember and return to promising states for further exploration, suggests that similar state-archiving and goal-directed exploration mechanisms would be essential for tackling games like Solomon's Club.

In Tetris, PPO generally outperformed other agents across various starting levels, consistently achieving higher mean episode rewards. However, despite indications that additional training steps could further enhance its performance, the overall results remain low. This aligns with previous findings [72], highlighting the persistent challenge of learning to play Tetris directly from raw pixel inputs.

## V. Conclusion

This work introduces the Game Boy Learning Environment (GLE), a novel platform for evaluating and developing deep reinforcement learning algorithms using classic Nintendo Game Boy games. The GLE offers a unique and challenging set of 11 game environments, each demanding long-term planning, strategic foresight, and hierarchical decision-making. Our platform innovates by providing both ultimate tasks, representing the game's final goal (called ultimate task), and subtasks, which offer intermediate challenges. To demonstrate the GLE's potential as a benchmark, we evaluated several agents using PPO, DQN, SAC and TD3 algorithms across subtasks for each game.

Our results demonstrate that agent performance varied significantly across game categories. PPO emerged as the most effective agent for role-playing games, while DQN excelled at completing levels in platform games. However, SAC outperformed other agents in defeating bosses within platform games. These findings highlight the diverse challenges presented by these environments, underscoring the limitations of current reinforcement learning techniques and the need for further advancements in the field. The results emphasize the value of the GLE as a rigorous testing ground for reinforcement learning algorithms, pushing the boundaries of what is possible in game-based AI research and opening new avenues for future work in this domain.

Concluding, we believe that as artificial intelligence systems are transitioning from static, task-specific models to more dynamic, general-purpose agents, games (such as Pokemon or The Legend of Zelda) requiring exploration, NPC interaction, natural language understanding, combat, trading, and item management could serve as valuable testbeds. Mastering such complex, open-ended challenges could drive progress towards more versatile and capable AI systems.

GLE is an evolving project with potential for expansion in environments, tasks, and reward functions. While facing challenges in comprehensive RAM address documentation, this limitation presents opportunities for community-driven growth. Future work will focus on leveraging collective expertise to enhance the GLE ecosystem, with a streamlined process for adding new games and environments detailed in the *Supplementary Material*.

## References


[1] Towers, M., Terry, J., Kwiatkowski, A., Balis, J., Cola, G., Deleu, T., Goulão, M., Kallinteris, A., KG, A., Krimmel, M., Perez-Vicente, R., Pierré, A., Schulhoff, S., Tai, J., Shen, A. & Younis, O. Gymnasium. (Zenodo,2023,3), https://zenodo.org/record/8127025

[2] Kent, S. The Ultimate History of Video Games, Volume 1: From Pong to Pokemon and Beyond... the Story Behind the Craze That Touched Our Lives and Changed the World. (Crown,2010)

[3] Bellemare, M., Naddaf, Y., Veness, J. & Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal Of Artificial Intelligence Research*. **47** pp. 253-279 (2013)

[4] Mei, Q., Xie, Y., Yuan, W. & Jackson, M. A Turing test of whether AI chatbots are behaviorally similar to humans. *Proceedings Of The National Academy Of Sciences*. **121**, e2313925121 (2024)

[5] Moravec, H. Mind children: The future of robot and human intelligence. (Harvard University Press,1988)

[6] Jiang, A., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D., Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L. & Others Mistral 7B. *ArXiv Preprint ArXiv:2310.06825*. (2023)

[7] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S. & Others Llama 2: Open foundation and fine-tuned chat models. *ArXiv Preprint ArXiv:2307.09288*. (2023)

[8] Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J., Yu, J., Soricut, R., Schalkwyk, J., Dai, A., Hauth, A. & Others Gemini: a family of highly capable multimodal models. *ArXiv Preprint ArXiv:2312.11805*. (2023)

[9] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M. & Sutskever, I. Zero-shot text-to-image generation. *International Conference On Machine Learning*. pp. 8821-8831 (2021)

[10] Oppenlaender, J. The creativity of text-to-image generation. *Proceedings Of The 25th International Academic Mindtrek Conference*. pp. 192-202 (2022)

[11] Ko, H., Park, G., Jeon, H., Jo, J., Kim, J. & Seo, J. Large-scale text-to-image generation models for visual artists' creative works. *Proceedings Of The 28th International Conference On Intelligent User Interfaces*. pp. 919-933 (2023)

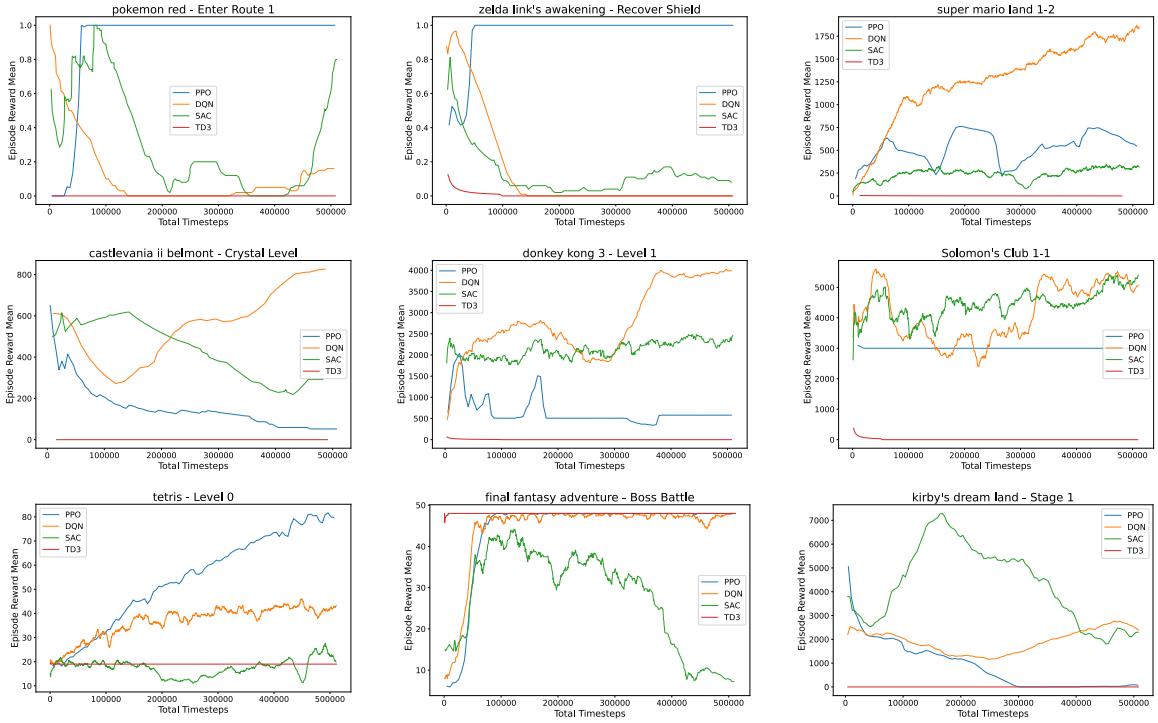


Fig. 2. The plots delineate the training progression based on the average reward accumulated on the episodes taken between each update. The rewards shown in the plots are defined as follows: for Pokemon Red and The Legend of Zelda: Link's Awakening, a reward of 1 is assigned upon successful task completion; Final Fantasy uses the remaining boss health as the reward signal; Super Mario Land employs the distance paved towards the goal as the reward. All other environments use the in-game score as the reward metric.


[12] Fenwick, A. & Molnar, G. The importance of humanizing AI: using a behavioral lens to bridge the gaps between humans and machines. *Discover Artificial Intelligence*. **2**, 14 (2022,8,25), https://doi.org/10.1007/s44163-022-00030-8

[13] Cárdenas-García, J. Info-Autopoiesis and the Limits of Artificial General Intelligence. *Computers*. **12** (2023), https://www.mdpi.com/2073-431X/12/5/102

[14] Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A. & Others Competition-level code generation with alphacode. *Science*. **378**, 1092-1097 (2022)

[15] Shen, C., Yao, V. & Liu, Y. Everybody Compose: Deep Beats To Music. *Proceedings Of The 14th Conference On ACM Multimedia Systems*. pp. 353-357 (2023), https://doi.org/10.1145/3587819.3592542

[16] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T. & Others Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *ArXiv Preprint ArXiv:1712.01815*. (2017)

[17] Silver, D., Huang, A., Maddison, C., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. & Others Mastering the game of Go with deep neural networks and tree search. *Nature*. **529**, 484-489 (2016)

[18] Perolat, J., De Vylder, B., Hennes, D., Tarassov, E., Strub, F., Boer, V., Muller, P., Connor, J., Burch, N., Anthony, T. & Others Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*. **378**, 990-996 (2022)

[19] Yang, S., Barlow, M., Townsend, T., Liu, X., Samarasinghe, D., Lakshika, E., Moy, G., Lynar, T. & Turnbull, B. Reinforcement Learning Agents Playing Ticket to Ride–a Complex Imperfect Information Board Game with Delayed Rewards. *IEEE Access*. (2023)

[20] Van Hasselt, H., Guez, A. & Silver, D. Deep reinforcement learning with double q-learning. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **30** (2016)

[21] Vinyals, O., Babuschkin, I., Czarnecki, W., Mathieu, M., Dudzik, A., Chung, J., Choi, D., Powell, R., Ewalds, T., Georgiev, P. & Others Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*. **575**, 350-354 (2019)

[22] Yeh, C., Li, V., Datta, R., Arroyo, J., Christianson, N., Zhang, C., Chen, Y., Hosseini, M., Golmohammadi, A., Shi, Y. & Others SustainGym: Reinforcement Learning Environments for Sustainable Energy Systems. *Advances In Neural Information Processing Systems*. **36** (2024)

[23] Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I. & Others Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *International Conference On Machine Learning*. pp. 1407-1416 (2018)

[24] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. *ArXiv Preprint ArXiv:1707.06347*. (2017)

[25] Adams, S., Arel, I., Bach, J., Coop, R., Furlan, R., Goertzel, B., Hall, J., Samsonovich, A., Scheutz, M., Schlesinger, M. & Others Mapping the landscape of human-level artificial general intelligence. *AI Magazine*. **33**, 25-42 (2012)

[26] Tesauro, G. & Others Temporal difference learning and TD-Gammon. *Communications Of The ACM*. **38**, 58-68 (1995)

[27] Schaeffer, J., Lake, R., Lu, P. & Bryant, M. Chinook the world man-machine checkers champion. *AI Magazine*. **17**, 21-21 (1996)

[28] Campbell, M., Hoane Jr, A. & Hsu, F. Deep blue. *Artificial Intelligence*. **134**, 57-83 (2002)

[29] Levinson, R. General game-playing and reinforcement learning. *Computational Intelligence*. **12**, 155-176 (1996)

[30] Genesereth, M., Love, N. & Pell, B. General game playing: Overview of the AAAI competition. *AI Magazine*. **26** pp. 62-62 (2005)

[31] Perez-Liebana, D., Samothrakis, S., Togelius, J., Schaul, T. & Lucas, S. General video game ai: Competition, challenges and opportunities. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **30** (2016)

[32] Perez-Liebana, D., Samothrakis, S., Togelius, J., Schaul, T., Lucas, S., Couëtoux, A., Lee, J., Lim, C. & Thompson, T. The 2014 general video game playing competition. *IEEE Transactions On Computational Intelligence And AI In Games*. **8** pp. 229-243 (2015)

[33] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. & Riedmiller, M. Playing atari with deep reinforcement learning. *ArXiv Preprint ArXiv:1312.5602*. (2013)

[34] Sumanas, M., Petronis, A., Bucinskas, V., Dzedzickis, A., Virzonis, D. & Morkvenaite-Vilkonciene, I. Deep Q-learning in robotics: Improvement of accuracy and repeatability. *Sensors*. **22**, 3911 (2022)

[35] Salimans, T. & Chen, R. Learning montezuma's revenge from a single demonstration. *ArXiv Preprint ArXiv:1812.03381*. (2018)

[36] Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. & Clune, J. Go-explore: a new approach for hard-exploration problems. *ArXiv Preprint ArXiv:1901.10995*. (2019)

[37] Johnson, M., Hofmann, K., Hutton, T. & Bignell, D. The Malmo Platform for Artificial Intelligence Experimentation.. *Ijcai*. **16** pp. 4246-4247 (2016)

[38] Tessler, C., Givony, S., Zahavy, T., Mankowitz, D. & Mannor, S. A deep hierarchical approach to lifelong learning in minecraft. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **31** (2017)

[39] Guss, W., Houghton, B., Topin, N., Wang, P., Codel, C., Veloso, M. & Salakhutdinov, R. Minerl: A large-scale dataset of minecraft demonstrations. *ArXiv Preprint ArXiv:1907.13440*. (2019)

[40] Guss, W., Castro, M., Devlin, S., Houghton, B., Kuno, N., Loomis, C., Milani, S., Mohanty, S., Nakata, K., Salakhutdinov, R. & Others The minerl 2020 competition on sample efficient reinforcement learning using human priors. *ArXiv Preprint ArXiv:2101.11071*. (2021)

[41] Hafner, D., Pasukonis, J., Ba, J. & Lillicrap, T. Mastering diverse domains through world models. *ArXiv Preprint ArXiv:2301.04104*. (2023)

[42] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T. & Others A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*. **362**, 1140-1144 (2018)

[43] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T. & Others Mastering atari, go, chess and shogi by planning with a learned model. *Nature*. **588**, 604-609 (2020)

[44] Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J. & Others Starcraft ii: A new challenge for reinforcement learning. *ArXiv Preprint ArXiv:1708.04782*. (2017)

[45] Wurman, P., Barrett, S., Kawamoto, K., MacGlashan, J., Subramanian, K., Walsh, T., Capobianco, R., Devlic, A., Eckert, F., Fuchs, F. & Others Outracing champion Gran Turismo drivers with deep reinforcement learning. *Nature*. **602**, 223-228 (2022)

[46] Berner, C., Brockman, G., Chan, B., Cheung, V., Dkebiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C. & Others Dota 2 with large scale deep reinforcement learning. *ArXiv Preprint ArXiv:1912.06680*. (2019)

[47] Pearce, T. & Zhu, J. Counter-strike deathmatch with large-scale behavioural cloning. *2022 IEEE Conference On Games (CoG)*. pp. 104-111 (2022)

[48] Jaderberg, M., Czarnecki, W., Dunning, I., Marris, L., Lever, G., Castaneda, A., Beattie, C., Rabinowitz, N., Morcos, A., Ruderman, A. & Others Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*. **364**, 859-865 (2019)

[49] Hausknecht, M., Ammanabrolu, P., Côté, M. & Yuan, X. Interactive fiction games: A colossal adventure. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **34** pp. 7903-7910 (2020)

[50] Narasimhan, K., Kulkarni, T. & Barzilay, R. Language understanding for text-based games using deep reinforcement learning. *ArXiv Preprint ArXiv:1506.08941*. (2015)

[51] Li, W., Devidze, R. & Fellenz, S. Learning to Play Text-Based Adventure Games with Maximum Entropy Reinforcement Learning. *Joint European Conference On Machine Learning And Knowledge Discovery In Databases*. pp. 39-54 (2023)

[52] (FAIR)†, M., Bakhtin, A., Brown, N., Dinan, E., Farina, G., Flaherty, C., Fried, D., Goff, A., Gray, J., Hu, H. & Others Human-level play in the game of Diplomacy by combining language models with strategic reasoning. *Science*. **378**, 1067-1074 (2022)

[53] Durante, Z., Sarkar, B., Gong, R., Taori, R., Noda, Y., Tang, P., Adeli, E., Lakshmikanth, S., Schulman, K., Milstein, A. & Others An interactive agent foundation model. *ArXiv Preprint ArXiv:2402.05929*. (2024)

[54] Tan, W., Ding, Z., Zhang, W., Li, B., Zhou, B., Yue, J., Xia, H., Jiang, J., Zheng, L., Xu, X. & Others Towards general computer control: A multimodal agent for red dead redemption ii as a case study. *ArXiv Preprint ArXiv:2403.03186*. (2024)

[55] Machado, M., Bellemare, M., Talvitie, E., Veness, J., Hausknecht, M. & Bowling, M. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal Of Artificial Intelligence Research*. **61** pp. 523-562 (2018)

[56] Bellemare, M., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D. & Munos, R. Unifying count-based exploration and intrinsic motivation. *Advances In Neural Information Processing Systems*. **29** (2016)

[57] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., Graves, A., Riedmiller, M., Fidjeland, A., Ostrovski, G. & Others Human-level control through deep reinforcement learning. *Nature*. **518**, 529-533 (2015)

[58] Laird, J. & Wray III, R. Cognitive architecture requirements for achieving AGI. *3d Conference On Artificial General Intelligence (AGI-2010)*. pp. 3-8 (2010)

[59] Kauten, C. Super Mario Bros for OpenAI Gym. (GitHub,2018), https://github.com/Kautenja/gym-super-mario-bros

[60] Ecoffet, A., Huizinga, J., Lehman, J., Stanley, K. & Clune, J. First return, then explore. *Nature*. **590**, 580-586 (2021)

[61] Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M. & Dormann, N. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal Of Machine Learning Research*. **22**, 1-8 (2021), http://jmlr.org/papers/v22/20-1364.html

[62] Yuan, A., Coenen, A., Reif, E. & Ippolito, D. Wordcraft: story writing with large language models. *27th International Conference On Intelligent User Interfaces*. pp. 841-852 (2022)

[63] Norström, L. Comparison of artificial intelligence algorithms for Pokémon battles. (2019)

[64] Silva, R. & Parpinelli, R. Playing the original game boy tetris using a real coded genetic algorithm. *2017 Brazilian Conference On Intelligent Systems (BRACIS)*. pp. 282-287 (2017)

[65] Branavan, S., Silver, D. & Barzilay, R. Learning to win by reading manuals in a monte-carlo framework. *Journal Of Artificial Intelligence Research*. **43** pp. 661-704 (2012)

[66] Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference On Machine Learning*. pp. 1861-1870 (2018)

[67] Fujimoto, S., Hoof, H. & Meger, D. Addressing function approximation error in actor-critic methods. *International Conference On Machine Learning*. pp. 1587-1596 (2018)

[68] Wang, X., Chen, Y. & Zhu, W. A survey on curriculum learning. *IEEE Transactions On Pattern Analysis And Machine Intelligence*. **44**, 4555-4576 (2021)

[69] Vithayathil Varghese, N. & Mahmoud, Q. A survey of multi-task deep reinforcement learning. *Electronics*. **9**, 1363 (2020)

[70] Gaina, R. & Stephenson, M. "Did you hear that?" Learning to play video games from audio cues. *2019 IEEE Conference On Games (CoG)*. pp. 1-4 (2019)

[71] Algorta, S. & Şimşek, Ö. The game of tetris in machine learning. *ArXiv Preprint ArXiv:1905.01652*. (2019)

[72] Liu, H. & Liu, L. Learn to play tetris with deep reinforcement learning. (2020)

[73] LeBlanc, D. & Lee, G. General Deep Reinforcement Learning in NES Games. *Canadian AI*. (2021)

[74] Osborn, J., Samuel, B., Summerville, A. & Mateas, M. Towards general rpg playing. *Proceedings Of The AAAI Conference On Artificial Intelligence And Interactive Digital Entertainment*. **13**, 92-98 (2017)

[75] Balloni, E., Mameli, M., Mancini, A. & Zingaretti, P. Deep Reinforced Navigation of Agents in 2D Platform Video Games. *Computer Graphics International Conference*. pp. 288-308 (2023)

[76] Oonishi, H. & Iima, H. Improving generalization ability in a puzzle game using reinforcement learning. *2017 IEEE Conference On Computational Intelligence And Games (CIG)*. pp. 232-239 (2017)

[77] Estermann, B., Lanzendörfer, L., Niedermayr, Y. & Wattenhofer, R. Puzzles: A benchmark for neural algorithmic reasoning. *Advances In Neural Information Processing Systems*. **37** pp. 127059-127098 (2024)

[78] Pleines, M., Addis, D., Rubinstein, D., Zimmer, F., Preuss, M. & Whidden, P. Pokemon Red via Reinforcement Learning. *ArXiv Preprint ArXiv:2502.19920*. (2025)

[79] Pathak, D., Agrawal, P., Efros, A. & Darrell, T. Curiosity-driven exploration by self-supervised prediction. *International Conference On Machine Learning*. pp. 2778-2787 (2017)

[80] Kulkarni, T., Narasimhan, K., Saeedi, A. & Tenenbaum, J. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances In Neural Information Processing Systems*. **29** (2016)

[81] Cobbe, K., Hesse, C., Hilton, J. & Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. *International Conference On Machine Learning*. pp. 2048-2056 (2020)