REINFORCEMENT LEARNING METHODS IN THE ROGUE GYM ENVIRONMENT

Nirav Adunuthula, Alexander Billups, Akalbir Singh Chandha
Department of Computer Science
University of Southern California
Los Angeles, CA 90007
{adunuthu,adb_376,akalbirs}@usc.edu

1 INTRODUCTION

Reinforcement learning (RL) is a machine learning technique where an agent learns to make optimal decisions by interacting with its environment without the benefit of human knowledge (Buffet et al., 2020). The objective of RL is to maximize the cumulative reward. However, RL problems pose several challenges:

- The reward signal is often sparse, delayed and noisy (Li, 2022). The agent must explore the environment to discover rewarding states.
- The environment may change dynamically, requiring the agent to continually adapt its policy (Yao et al., 2012).
- Balancing exploration and exploitation is difficult, as the agent must choose between acting greedily and acting to gain more information.

Generalization is a crucial aspect of RL, as it enables agents to adapt to new environments and perform well in different situations. RL agents that can generalize across various scenarios are more robust and versatile, making them suitable for real-world applications. One common example of generalization in RL is the ability of an agent to play different games or solve problems in changing environments.

The Rogue-Gym environment, a simple and classic style roguelike game, is specifically designed to challenge the generalization capabilities of RL agents. In this environment, an agent navigates through random dungeon rooms containing elements such as monsters, traps, doors, keys, and potions. The environment is a fully observable 2D grid world with RGB image observations, and the agent can perform discrete movement actions (up, down, left, right). Each episode involves the random generation of rooms, monsters, and objects, and the agent receives a sparse reward signal for collecting coins, moving to the next stage, and collecting the Amulet item. The environment is episodic, with variable episode lengths. We use the diverse dungeon structure to assess the generalization ability of two RL agents and attempt to improve upon the original authors' performance(Kanagawa & Kaneko, 2019).

In the Rogue-Gym environment, traditional RL methods like Proximal Policy Optimization (PPO) often overfit to the training environment and struggle to generalize to new scenarios. To address this, hierarchical RL and Transformers are promising approaches. Hierarchical RL breaks down tasks into high-level goals and low-level actions, potentially enhancing the agent's adaptability to diverse structures in Rogue-Gym. Transformers, successful in recent RL applications, can capture long-range dependencies in sequential data, useful in Rogue-Gym where decisions are based on various element configurations and interactions over time. The combination of hierarchical RL and Transformers could potentially improve the generalization ability of RL agents and mitigate the overfitting problem observed with traditional PPO agents in Rogue-Gym(Fahad Mon et al., 2023).

2 BACKGROUND

Reinforcement Learning. We define the RL problem as a Markov Decision Process $M = (S, A, R, P, t)$. At each timestep t, the agent uses a policy $\pi(a_t|s_t)$ to choose an action $a_t \in A$

given a state $s \in S$. The agent performs the action and the environment generates a reward r_t based on the action and the state. We define the reward for an episode as the total reward or return given by $\sum_{t=0}^{T} R(s_t)$. The goal of RL is to learn a policy that maximizes the reward.

Model Based Reinforcement Learning. This broadly refers to approaches that estimate the transition function of the environment to assist in planning. This differs from model-free RL which traditionally learns the policy by interacting with and exploring the environment. These models can also be used to generate samples for traditional model-free RL algorithms.

Offline RL. In this formulation, the agent is given access to an offline dataset defined as $D = (s, a, s', r)$. The model is entirely trained on the dataset. D, and does not see the actual environment. This scalable data driven approach is appealing, especially since models trained on huge amounts of data like GPT and image models like DALLE have shown high generalization capabilities (Bubeck et al., 2023). Offline RL could make powerful decision making engines (Levine et al., 2020). To date, offline RL has shown promise in Robotics, healthcare, autonomous driving, and healthcare due to the expensive and or prohibitive nature of online interaction (Li et al., 2023).

Transformers. The Transformer, as proposed by Vaswani et al. (2023)Vaswani et al. in 2017, is recognized as a highly effective and scalable neural network for the modeling of sequential data. Its core concept involves the incorporation of a self-attention mechanism, providing the model with the ability to efficiently capture global dependencies within extended sequences. It has shown better performance than CNNs and RNNs across many learning tasks, and could potentially perform better at the RL problem of state representation and dynamical modeling(Li et al., 2023).

3 Related Work

Environment. Yuji Kanagawa and Tomoyuki Kaneko propose ”Rogue-Gym: A New Challenge for Generalization in Reinforcement Learning”, the environment we use for this study. The authors conducted experiments using Proximal Policy Optimization, with and without enhancements for generalization. The results showed that some enhancements believed to be effective failed to mitigate the overfitting in Rogue-Gym, although others slightly improved the generalization ability. The authors conclude that Rogue-Gym presents a novel and challenging domain for further studies on the generalization ability of RL agents(Kanagawa & Kaneko, 2019).

Hierarchical RL. Shao et al. (2019)Shao et al.(2019) provide a comprehensive survey of deep RL applications in video games, highlighting the benefits of Hierarchical Reinforcement Learning (HRRL) in decomposing tasks into simpler subtasks, speeding up training, and improving sample efficiency. Adapting HRRL methods from Pinto & Coutinho (2019) Pinto et al.(2018) and Wang et al. (2020)Wang et al.(2021), the authors apply these techniques to the Rogue Gym game. Pinto et al.'s method uses Q-learning in the FightingICE platform to assess the environment and dictate the Monte Carlo Tree Search for the best move. Wang et al.'s method focuses on interactions between high and low-level policies in an ant maze, gather, and push environment. By adapting these HRRL methods, we aim to maximize rewards in the Rogue Gym game, addressing its multiple reward mechanisms and complex environment.

Transformers in RL. The intersection of Transformers and RL presents a burgeoning field with the potential to improve decision-making in complex environments. We apply the Decision Transformer to Rogue Gym (Chen et al., 2021). This approach outputs optimal actions by auto-regressively modeling state-action-reward trajectories, defined as $\tau = (\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2, ...\hat{R}_T, s_T, a_T)$ where $\hat{R}_t = \sum_{t'=t}^{T} r(s_{t'}, a_{t'})$ is the return-to-go. DT is shown to perform effective long term credit assignment and be better in sparse reward settings than TD learning algorithms. The Trajectory Transformer (Janner et al., 2021) similarly models RL as a sequence prediction problem, and the authors show that it can perform imitation learning, goal-conditioned RL and offline RL. Offline models struggle like other supervised learning approaches with limited datasets that don't have large distribution coverage (Li et al., 2023). This was addressed in the Bootstrapped Transformer(Wang et al., 2022) which uses the learned model to self-generate offline data and boost the sequence model training.

4 Problem Formulation

Rogue-Gym formulates the roguelike game Rogue as a partially observable Markov decision process (POMDP).

State Space (S): Map layout, entity locations, inventories, statuses. The agent receives partial observation $o \in O$ as input where O is RGB images/ASCII of size H×W representing the agent's current view .

Action Space (A): The discrete set of allowable actions $a \in A$ consists of movements in the 8 directions, search, inventory, and no-op

Transition Function (P): $P(s'|s, a)$ determines the next full state $s' \in S$ given the current state $s \in S$ and action $a \in A$. This encapsulates all the Rogue game rules, mechanics and dynamics. The transitions are stochastic due to the procedural generation of each level's map layout, monster behaviors, combat outcomes etc.

Reward Function (R): The reward function $R(s, a, s') \rightarrow r \in R$ outputs a scalar reward r given the state transition (s, a, s'). Collecting coins is +1, Taking a staircase is +50, and picking up the Amulet item is +100. This makes rewards extremely sparse.

Objective: Learn a stochastic policy $\pi(a|o)$ that maps observations $o \in O$ to action probabilities to maximize expected cumulative episodic return. As observations do not fully convey the underlying state, this requires generalization across a distribution of unseen maps and entity configurations.

Configuration: Agents were trained on randomized seeds between [0,40] for each episode. There were no enemies and there was a set dungeon style shared between each seed.

Evaluation: The trained agents are evaluated on their ability to generalize to unseen levels. This is measured by the mean episodic return over multiple episodes on new level configurations.

5 Methodology

5.1 Proximal Policy Optimization (PPO) Agents

Proximal Policy Optimization (PPO) (Schulman et al., 2017) is a policy optimization method that uses a surrogate objective function to improve the policy while ensuring that the new policy does not deviate too much from the old one. This balance between exploration and exploitation makes PPO a popular choice for training RL agents.

CNN+PPO. The Nature-CNN agent uses a simple 3-layer convolutional architecture proposed by (Mnih et al., 2015)Mnih et al., 2015. In contrast, the more complex IMPALA-CNN Agent (Espeholt et al., 2018) with 15 CNN layers and 3 residual connections is part of the IMPALA framework and has shown better generalization across training sets compared to Nature-CNN, despite both being trained with PPO.

VAE Agent + PPO. The Variational Autoencoder (VAE)(Kingma & Welling, 2019) Agent is a generative model that learns a latent representation of the input data and passes the compressed representation to the PPO model. This approach is expected to help the agent generalize better because the VAE's latent representation can capture more general features of the game state. The VAE is trained to reconstruct the game state from its latent representation, and the PPO agent is trained to maximize the expected reward given this latent representation

5.2 Hierarchical Reinforcement Learning

Q Learning + Monte Carlo Search Approach. We adapted the Hierarchical Reinforcement Learning approach from(Pinto & Coutinho, 2019) Pinto et al. (2018) to a roguelike game environment. Our Q-learning function approximates Q-values using binary features indicative of key aspects in the game state: presence in dungeon/passage, nearby doors/stairs/gold. Weights for these features are initialized randomly between 0-1 for exploration. An ϵ -greedy policy balances exploiting learned knowledge vs. trying new actions. The best state from this policy seeds a Monte Carlo Tree Search to further explore possible actions and outcomes to refine the agent's understanding and decision-making. By tailoring the hierarchical methodology to the roguelike domain, we aim to accelerate learning in this complex environment.

High and Low Level Policy Approach. This approach adapts interactive influence-based hierarchical reinforcement learning (I2HRL) (Wang et al., 2020) to the Rogue Gym environment with its two reward functions - advancing deeper into dungeons and collecting gold. It utilizes separate high-level and low-level policies with LSTM architectures to select actions. The high-level policy focuses on navigation and next level advancement. It calculates action probabilities to maximize immediate rewards. A gold check determines if the low-level policy should be activated to maximize gold collection in addition to advancement. If no gold is present, only the high-level policy selects actions and gets updated. The low-level policy is activated when gold is detected to choose gold-maximizing actions, before both policies update based on observed rewards.

By decomposing the task into complementary high-level navigation and low-level gold collection policies, this hierarchical approach aims to accelerate learning in the complex Rogue Gym environment. Also we hope the separate policies could allow for focused optimization on the two distinct reward mechanisms. Two additional integration approaches were explored between high-level and low-level policies. The first, lacking a gold-check mechanism, resulted in no action in environments without a gold objective. The second approach aimed to separate explicit movement from information-seeking actions, but faced challenges as the lower-level policy tended to continuously seek information, disrupting planned movements. Both approaches posed difficulties in aligning with the dual reward functions outlined in the paper, emphasizing the nuanced considerations required in hierarchical RL frameworks.

5.3 Decision Transformer

We use the output of the PPO models from the original paper to generate a dataset of episodes that we could use to perform offline RL. The code for this can be seen here. (https://github.com/rogue-agents-sc/rogue-gym-agents) We generated short and long-term trajectories of 100 and 500 steps and used these to train the decision transformer. We wanted to see if adjusting the distribution of high reward episodes resulted in improved performance in the model, so we additionally conditioned the sample episode generations on whether they had a cumulative reward over the value 100. This value equated to the agent going down one stairway in the rogue gym environment. An episode that reached a reward of 100 in a small number of steps should have taken a large number of beneficial actions for the given states, potentially improving the model's understanding of the environment.

6 Results

In this section, we present the results of our experiments with various RL agents in the Rogue-Gym environment. Our study aimed to explore different architectures and methods to improve the agents' performance in this challenging roguelike game setting.

6.1 PPO Agents

We compared the performance of three Proximal Policy Optimization (PPO) agents—Nature-CNN, IMPALA-CNN, and VAE-PPO—each with distinct features and learning capabilities.

1 depicts the mean reward trends among three agents in Rogue-Gym. The Nature-CNN agent, simpler but steady, scores lower than others. IMPALA-CNN, complex with 15 layers, excels, showing adaptability to new levels. VAE-PPO, using a Variational Autoencoder, offers diverse learning but lower rewards. Compared with the original paper, IMPALA-CNN outperforms Nature-CNN, showcasing the benefits of complex architectures. These insights deepen our understanding of neural networks in game environments like Rogue-Gym, shedding light on their effectiveness compared to benchmarks in the field.

6.2 Hierarchical Reinforcement Learning

When comparing our results in 6.2 using HRRL versus the prior agents, it lagged behind drastically. Over the span of 100 episodes, both the HRRL methods failed to reach a reward mean score of 0

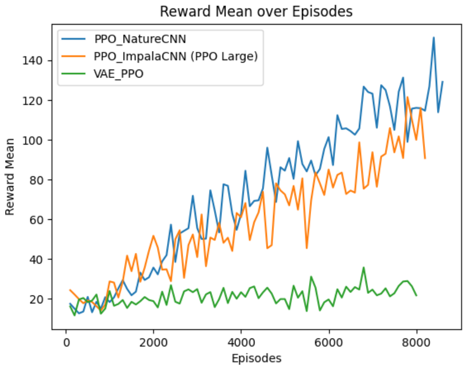
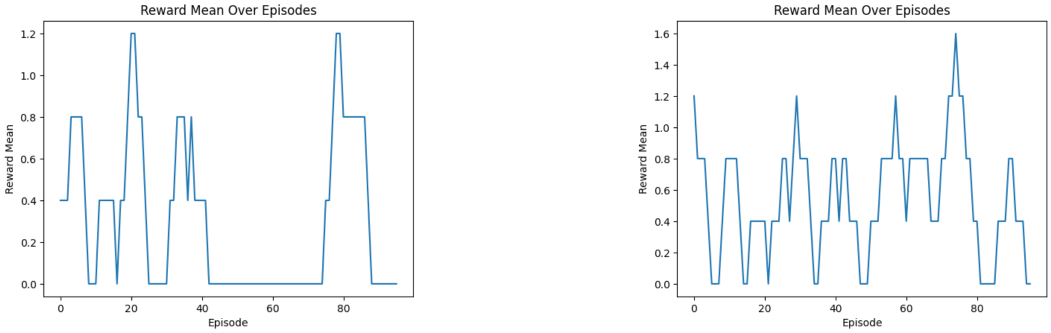


Figure 1: Results of the PPO Agents after replication



(a) High+Low Level Policy

(b) Q learning and MCTS

Figure 2: HRRL Results

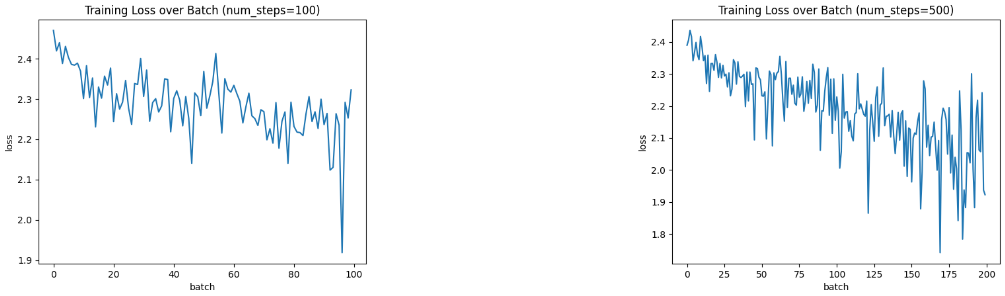
despite its counterparts reaching 17.5(Nature CNN), 24.4(Impala CNN), and 16.2(VAE PPO) during that same time span. It could be counted as a success that at the very least both methods have the capability of achieving one of the goals outlined in the paper which is to collect gold within the dungeons. However, the biggest detriment to these two models is its lack of ability to explore past the starting dungeon that it begins in. As the biggest reward presented in the environment, by not having the ability to execute finding and going downstairs, these models fail at reaching the same reward means posed by the pre-existing agents.

Examining the test results, both models exhibited clustered action states for both ”no operation” and ”search” actions. Despite offering no added benefit in expanding the model’s view from the previous state, these actions remained constant. Although removing ”no operation” improved the chances of obtaining a reward, eliminating ”search” is not feasible due to its intrinsic value in certain situations. The models struggle to strike a balance between operating in a generalized environment and actively seeking information in a space with limited pointers.

6.3 Decision Transformer

Despite running many experiments for the decision transformer over different dataset distributions and different hyperparameters, we were unable to see any learning in the decision transformer model. As you can see in 6.3, the loss for the decision transformer remains effectively the same throughout and they are not significantly different for 100 vs 500 step episodes.

We believe this is due to programming error, but this could also be due to limitations we faced and the nature of our environment. Our state was stored as a string of the maze represented with ASCII symbols, and the primary part of the state that updated was the character location. Because the



(a) Loss for 100 step eps

(b) Loss for 500 step eps

Figure 3: Decision Transformer Training Loss

environment remained mostly the same, the model might not have been able to learn how to use the character location to inform its predicted action. This was shown in evaluation where the model would always predict the same value even after the character's location changed. Perhaps modifying the state input to focus on player location and performing a convolution over the input image like the PPO algorithms might result in better performance.

We had initial experiments where we prompted GPT 4 to take in a Rogue Gym state and return an action, but the model would struggle to keep a consistent state. This was especially the case when the player character would go on top of the staircase. Here the model should predict the downstairs action, but in our testing, GPT would either move the downstairs symbol (%) to another location or forget that the character was on the stairs. It's unlikely GPT would have seen much Rogue environment data, so it's unsurprising that it fails these tests.

7 Conclusion and Future Work

Our comprehensive examination of reinforcement learning agents within the challenging Rogue-Gym environment has produced valuable insights into their diverse performances and limitations. The comparison of Proximal Policy Optimization (PPO) agents, including Nature-CNN, IMPALA-CNN, and VAE-PPO, highlights the nuanced impact of architectural choices on generalization and adaptability to unseen levels. The results demonstrate that the more complex architecture leads to superior generalization and higher mean rewards, aligning with the notion that intricate structures enhance adaptability in unfamiliar environments. In contrast, our foray into HRRL and Decision Transformers exposed notable challenges. While HRRL succeeded in gold collection, exploration limitations hindered it from matching pre-existing agents' ultimate rewards. Furthermore, the Decision Transformer faced unexpected challenges, possibly due to its offline nature. To enhance its performance, future iterations may focus on refining state representations by emphasizing player location to encourage more movement in the environment and to the goal along with adopting convolutional methods. Despite being unsuccessful in matching the pre-existing agents, our results contribute positive insights towards the direction of reinforcement learning in a dynamic, generalized game environment, emphasizing the need for complex and expanded architecture to fully encapsulate the unknown present.

For Future work, Foundation Models like GPT and DALLE can enhance RL performance by generating contextual information about the environment to augment the state representation and by assisting in complex decision-making tasks. These models contain pre-existing knowledge that can improve agent's generalization ability and aid in designing more sophisticated reward functions. Future work could focus on integrating AI language models with reinforcement learning frameworks to improve communication between agents and the environment, ultimately enhancing the agent's ability to learn and make informed decisions in dynamic and diverse scenarios.

References

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

Olivier Buffet, Olivier Pietquin, and Paul Weng. Reinforcement learning, 2020.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures, 2018.

Bisni Fahad Mon, Asma Wasfi, Mohammad Hayajneh, Ahmad Slim, and Najah Abu Ali. Reinforcement learning in education: A literature review. *Informatics*, 10(3), 2023. ISSN 2227-9709. doi: 10.3390/informatics10030074. URL https://www.mdpi.com/2227-9709/10/3/74.

Michael Janner, Qiyang Li, and Sergey Levine. Reinforcement learning as one big sequence modeling problem. In *ICML 2021 Workshop on Unsupervised Reinforcement Learning*, 2021. URL https://openreview.net/forum?id=AfDCOISXx1T.

Yuji Kanagawa and Tomoyuki Kaneko. Rogue-gym: A new challenge for generalization in reinforcement learning, 2019.

Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392, 2019. ISSN 1935-8245. doi: 10.1561/2200000056. URL http://dx.doi.org/10.1561/2200000056.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.

Wenzhe Li, Hao Luo, Zichuan Lin, Chongjie Zhang, Zongqing Lu, and Deheng Ye. A survey on transformers in reinforcement learning, 2023.

Yuxi Li. Reinforcement learning in practice: Opportunities and challenges, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015. URL https://api.semanticscholar.org/CorpusID:205242740.

Ivan J. P. Pinto and Luciano Reis Coutinho. Hierarchical reinforcement learning with monte carlo tree search in computer fighting game. *IEEE Transactions on Games*, 11:290–295, 2019. URL https://api.semanticscholar.org/CorpusID:64720661.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

Kun Shao, Zhentao Tang, Yuanheng Zhu, Nannan Li, and Dongbin Zhao. A survey of deep reinforcement learning in video games, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Kerong Wang, Hanye Zhao, Xufang Luo, Kan Ren, Weinan Zhang, and Dongsheng Li. Bootstrapped transformer for offline reinforcement learning, 2022.

Rundong Wang, Runsheng Yu, Bo An, and Zinovi Rabinovich. I²hrl: Interactive influence-based hierarchical reinforcement learning. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 3131–3138. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/433. URL `https://doi.org/10.24963/ijcai.2020/433`. Main track.

Hengshuai Yao, Davood Rafiei, and Rich Sutton. A study of temporal citation count prediction using reinforcement learning. `https://hengshuaiyao.github.io/papers/citation.pdf`, 2012. Accessed: 2023-12-9.

A Appendix

The pre-existing agents are trained using the Proximal Policy Optimization (PPO) algorithm.

Table 1: Training Parameters for PPO Agent

Training Parameter Name	Training Parameter Value
Number of workers	32
Number of steps	125
Value loss weight	0.5
Entropy weight	0.01
GAE tau	0.95
Use of GAE	True
PPO minibatch size	400
PPO clip	0.1
Learning rate decay	False

The training is performed using either CNN or ResNet architectures for the policy network. The VAE agent uses the latent representation from the VAE as input to the policy network instead.

Table 2: Environment Configuration

Environment Parameter	Environment Parameter Values
Width	32
Height	16
Hide Dungeon	True
Dungeon style	”rogue”
Room number in x-axis	2
Room number in y-axis	2
Enemies	None

Table 3: Environment Configuration

Evaluation. The trained agents are evaluated for their generalization capabilities on 500 randomly generated unseen levels from Rogue-Gym. The metric reported is the mean episodic return over 5 random seeds. The evaluation environment is set to use a stair reward of 100.0 and a maximum of 500 steps per episode.

B Limitations

The Rogue Gym project presents a notable limitation in the absence of a Python API, as highlighted in the paper. This lack of an API restricts the development of diverse reinforcement learning (RL) agents beyond the predefined ones documented in the paper. Additionally, the code for agents trained in the paper lacks adaptability for use outside the predefined RL methods covered in the study. To address this limitation, a crucial step was taken in the Hierarchical Reinforcement Learning (HRRL) framework to overcome this hurdle. A method was devised to convert the visual elements of the Rogue Gym environment, generated by the existing code, into tangible data. This transformation enables the training and testing of RL methods and practices beyond the confines of the original paper, thereby expanding the applicability of the Rogue Gym environment for future research and experimentation.

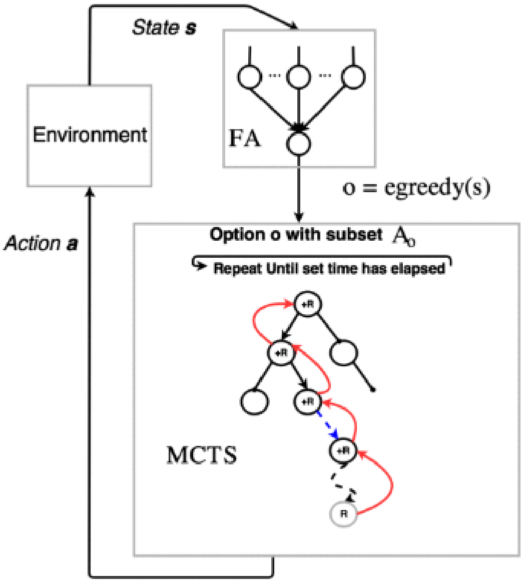
Another limitation was our lack of computing power. We weren't able to reach the millions of frames necessary to train and test the agents like proposed in the model. Thus, we decided to focus on maximizing the performance of our model in smaller episodic caps in the hundreds and smaller thousands.

C Code

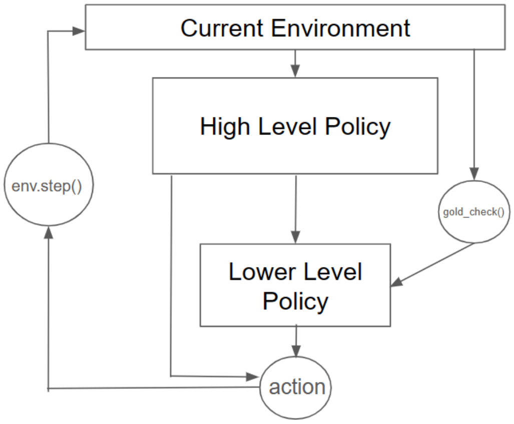
Code for the paper can be found in the organization located here: https://github.com/orgs/rogue-agents-sc/repositories

D Visual

Q Learning and Monte Carlo Search Method From (Pinto & Coutinho, 2019)



HRRL with a High and Low Level Policy Loosely Based on Wang et al. (2020)Wang et al.(2021)


Current Environment
High Level Policy
env.step()
gold_check()
Lower Level Policy
action