Semantic State Representation for Reinforcement Learning

Erez Schwartz
Technion
erezschwartz@campus.technion.ac.il

Guy Tennenholtz
Technion
sguyt@campus.technion.ac.il

Chen Tessler
Technion
chen.tessler@campus.technion.ac.il

Shie Mannor
Technion
shiemannor@gmail.com

Abstract

Recent advances in reinforcement learning have shown its potential to tackle complex real-life tasks. However, as the task's dimensionality increases, reinforcement learning methods tend to struggle. To overcome this, we explore methods for representing the semantic information embedded in the state. While previous methods focused on information in its raw form (e.g., raw visual input), we propose representing the state as natural language. Language can represent complex scenarios and concepts, making it a favorable candidate for representation. Empirical evidence, within the domain of ViZDoom, suggests that natural language based agents are more robust, converge faster and perform better than vision based agents, showing the benefit of using natural language representations for reinforcement learning.

1 Introduction

Deep learning based algorithms use neural networks in order to learn feature representations that are good for solving high dimensional machine learning (ML) tasks. Reinforcement learning (RL) is a subfield of ML that has been greatly affected by the use of deep neural networks as universal function approximators [Csáji, 2001]. These deep neural networks are used in RL to estimate value functions, state-action value functions, policy mappings, next-state predictions, rewards, and more [Mnih et al., 2015; Schulman et al., 2017] using their representation power, thus combating the "curse of dimensionality" [Powell, 2007]. For the purpose of this paper we define a **semantic representation** of a state as one that reflects its meaning as it is understood by an expert. The semantic representation of a state should thus be paired with a reliable and computationally efficient method for extracting information from it. Previous success in RL has mainly focused on representing the state in its raw form (e.g., visual input in Atari-based games [Mnih et al., 2015]). In this work, we challenge current representation techniques and suggest to represent the state using natural language, similar to the way we, as humans, summarize and transfer information efficiently from one to the other [Sapir, 2004]. In this work we assume a state can be described using natural language sentences. Fig. 1 depicts a block-diagram of the basic framework we rely on. In Section 3 we compare NLP representations with their alternatives. Our results suggest that representation of the state using natural language can achieve better performance, even on difficult tasks, or tasks in which the description of the state is saturated with task-nuisances [Achille and Soatto, 2018].

NeurIPS 2020 Workshop on Object Representations for Learning and Reasoning, Vancouver, Canada.

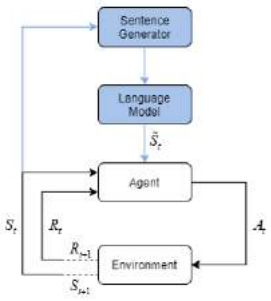

Figure 1: Block diagram for using language-based state representation RL.

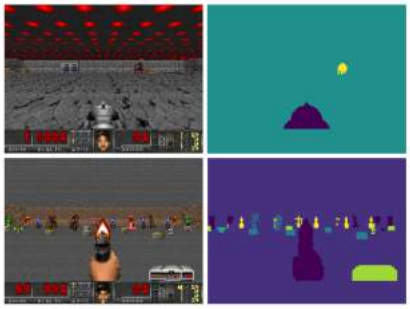

Figure 2: Raw visual inputs and their incorresponding semantic segmentation in the VizDoom enviornment.

Figure 3: Examples of semantic state granularity in natural language.

2 Natural Language State Representation in the Doom Environment

The ViZDoom environment [Kempka et al., 2016] is a 3D world with a relatively realistic physics model and significantly more realistic visual input than Atari 2600 games [Mnih et al., 2015]. There, an agent must effectively perceive, interpret, and learn the 3D world in order to make tactical and strategic decisions of where to go and how to act. There are two types of visual representations that are provided by the environment: (1) *raw visual inputs*, also the most commonly used form, in which the state is represented by an image from a first person view of the agent, and (2) a *semantic segmentation map* based on the positions and labels of all objects and creatures in the vicinity of the agent. An example of such visual representations in VizDoom is presented in Fig. 2. Note that semantic segmentation maps are a compact form for representing what a designer believes to be useful features of the state.

2.1 Natural Language Generator

To simulate natural language based captioning of the state, we constructed a semantic natural language parser. To guarantee a fair comparison against visual representations, we constructed descriptions based on two criteria: limiting information and ambiguity.

2.1.1 Limiting Information

In real world applications, natural language can be efficiently used to describe non-spatial attributes that would be hard otherwise. Some examples include *"The enemy is too close, watch out!"* and *"An enemy is walking towards you quickly."*. While we believe this aspect of natural language is one of its greater merits, for fair comparison, we limited the expressiveness of our parser to the spatial information present in the semantic segmentation maps. ViZDoom's semantic segmentation maps were used as the core element for generating our natural language descriptions. Each state of the environment was converted into a sentence based on positions and labels of objects in the frame, as presented in these maps. To implement this, the screen was divided into several vertical and horizontal patches, as depicted in Fig. 3. These patches describe relational aspects of the state, such as distance of objects and their direction with respect to the agent's point of view. In each patch, objects were counted, and a sentence description was constructed based on this information.

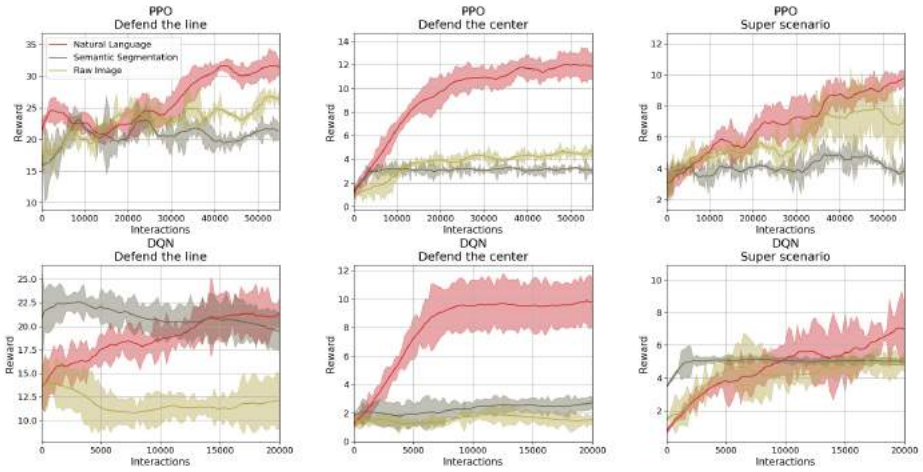

Figure 4: Comparison of representation methods on the different VizDoom scenarios using DQN and PPO agents. X and Y axes represent the number of iterations and cumulative reward, respectively.

2.1.2 Ambiguity

A key element of natural language is ambiguity, i.e., there is more than one way to describe a state. For example, all of the following sentences contain information of an agent shooting and missing a target: "*You shot the wall.*", "*The player shot and missed.*", and "*The monster dodged your bullet.*". Coping with ambiguity in natural language is a cornerstone for Explainable Artificial Intelligence [Samek et al., 2017], increasing interpretability and transparency of black-box deep learning models. To simulate ambiguity, we constructed ten distinct sentence generators that described each patch of every game frame in a different manner. More specifically, at each step and for every patch, a random parser was sampled, describing the given patch, forming the final state representation. The parsers varied in word usage and sentence structure, allowing for a wide range of descriptions for every state.

3 Experiments

We tested natural language state representation using our parser against visual-based representations on several tasks, with varying difficulty. In these tasks, the agent can navigate, shoot, and collect items such as weapons and medipacks. The agent obtains a positive reward when it kills the various enemies in each scenario. The different scenarios include a scenario in which the agent must take cover from inbound fireballs (Defend the Line), a scenario in which the agent must defend itself from charging enemies (Defend the Center), and a "super" scenario, where a mixture of these scenarios was designed to challenge the agent. Our agent was implemented using a Convolutional Neural Network. The parsed state was converted into embedded representations of fixed length. The sentence length was defined by the longest sentence, whereas, shorter sentences were padded with zeros. In this work, we tested both a DQN and PPO based agent and compared the natural language state representation against the other representations mentioned earlier.

3.1 Performance

Results for DQN and PPO-based agents are presented in Fig. 4. Each plot depicts the average reward (across 5 seeds) of all representation methods. It can be seen that NLP representations outperform the visual ones, contrary to the fact that they contain the same information as the semantic segmentation maps. A more thorough analysis of the performance shows that although natural language eventually outperforms other representations, the difference and convergence time are domain dependent. Additionally, our results indicate that performance of different semantic representations are algorithm-dependent. Interestingly, raw visual inputs showed superiority over semantic segmentation maps when used by the PPO agent, whereas the reverse occurred with the DQN agent. This suggests that the effect of representations on performance is algorithm dependent.

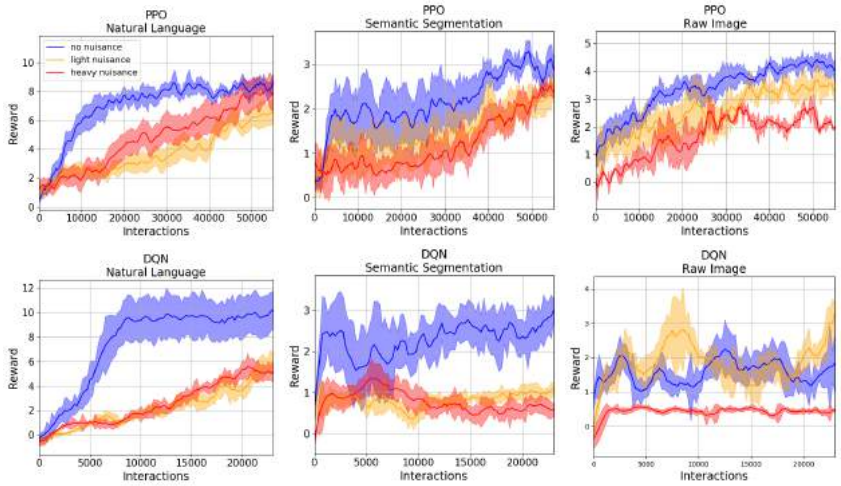

Figure 5: Robustness of representation with respect to nuisance amount, "Defend the Center" scenario.

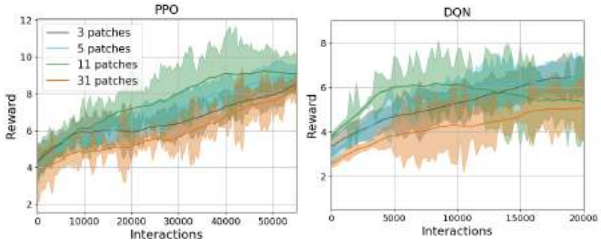

Figure 6: scores of NLP agent as a function of the number of patches. "Super" scenario

3.2 Robustness to Task Nuisances

Fig. 5 shows the effect of task-nuisances over the different representation types. There, a large amount of unnecessary objects were spawned in the level. It can be seen that inflation of the state space with task-nuisances impaired performance of all representations. These increased the state description length to over 250 words, whilst retaining the same amount of useful information. Nevertheless, in most scenarios, NLP representations maintained robustness to the applied noise, outperforming the vision based representations. The robustness of NLP representations to task-nuisances indicate their ability to summarize information well.

3.3 Discretization to Patches

Discretization of the frame to patches in the ViZDoom environment was not carried out for reasons of efficiency, but rather to devise the semantic language parser. Still, it is essential to understand its effect on the performance of the agent. To test the effect of discretization of the frame to patches, we conducted experiments with varying amounts of horizontal patches, ranging from 3 to 31 patches, in the extreme case. Our results, as depicted in Fig. 6, suggest that the amount of discretization has a negligible effect on the performance of the NLP-based agents.

4 Conclusion

We have shown that natural language representations help interpret the state of an agent, improving its overall performance. While this has worked well in the ViZDoom environment, it may not hold for all domains. Designers of RL algorithms should consider searching for a semantic representation that fits their needs. We also note that natural language representations can be used to augment specific parts of the state space. They can also act as a means to inject prior knowledge to the state. While

this work only takes a first step toward finding better semantic state representations, we believe the structure inherent in natural language can be considered a favorable candidate for achieving this goal.

References

Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 2018.

Balázs Csanád Csáji. Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24:48, 2001.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–8. IEEE, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*, 2017.

Edward Sapir. *Language: An introduction to the study of speech*. Courier Corporation, 2004.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.