Language is Power: Representing States Using Natural Language in Reinforcement Learning

Erez Schwartz, Guy Tennenholtz, Chen Tessler and Shie Mannor

Abstract

Recent advances in reinforcement learning have shown its potential to tackle complex real-life tasks. However, as the dimensionality of the task 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 to represent the state using 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

"Language is power, in ways more literal than most people think." – Julia Penelope

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].

The term representation is used differently in different contexts. 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]). This approach stems from the belief that neural networks (specifically convolutional networks) can extract meaningful features from complex inputs. 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].

The ability to associate states with natural language sentences that describe them is a hallmark of understanding representations for reinforcement learning. Humans use rich natural language to describe and communicate their visual perceptions, feelings, beliefs, strategies, and more. The semantics inherent to natural language carry knowledge and cues of complex types of content, including events, spatial relations, temporal relations, semantic roles, logical structures, support for inference and entailment, as well as predicates and arguments [Abend and Rappoport, 2017]. The expressive nature of language can thus act as an alternative semantic state representation.

Over the past few years, Natural Language Processing (NLP) has shown an acceleration in progress on a wide range of downstream applications ranging from Question Answering [Kumar et al., 2016] , to Natural Language Inference [Parikh et al., 2016] through Syntactic Parsing [Williams et al., 2018] . Recent work has shown the ability to learn flexible, hierarchical, contextualized representations, obtaining state-of-the-art results on various natural language processing tasks [Devlin et al., 2018]. A basic observation of our work is that natural language representations are also beneficial for solving problems in which natural language is not the underlying source of input. Moreover, our results indicate that natural language is a strong alternative to current complementary methods for semantic representations of a state.

In this work we assume a state can be described using natural language sentences. We use distributional embedding methods[1] in order to represent sentences, processed with a standard Convolutional Neural Network for feature extraction. Fig. 1 depicts a block-diagram of the basic framework we rely on – described in detail in Sections 2 and 4. We discuss possible semantic representations in Section 3,

[1]Distributional methods make use of the hypothesis that words which occur in a similar context tend to have similar meaning [Firth, 1957], i.e., the meaning of a word can be inferred from the distribution of words around it. For this reason, these methods are called distributional methods.

namely, raw visual inputs, semantic segmentation, and natural language representations. Then, in Section 5 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]. Moreover, we observe that NLP representations are more robust to transfer and changes in the environment. We conclude the paper with a short discussion and related work.

2 Preliminaries

2.1 Reinforcement Learning

We consider a discounted Markov Decision Process (MDP) [Sutton and Barto, 1998]. An MDP is defined as the tuple $(\mathcal{S}, \mathcal{P}, \mathcal{A}, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$ is a transition kernel, r is the reward function, and $\gamma \in [0, 1)$ is the discount factor.

The goal in RL is to learn a policy $\pi(s)$ which maximizes the reward-to-go $\sum_{t=0}^{\infty} \gamma^t r_t$. For a given policy, there are two common quantities which are used to estimate the performance: the value function $v^\pi(s) = \mathbb{E}^\pi[\sum_t \gamma^t r_t | s_0 = s]$ and action-value function $Q^\pi(s, a) = \mathbb{E}^\pi[\sum_t \gamma^t r_t | s_0 = s, a_0 = a]$. Intuitively, $v^\pi(s)$ represents how "good" a state s is and $A^\pi(s, a) = Q^\pi(s, a) - v^\pi(s)$ represents how the agent would improve by playing action a at state s when compared to the current policy π. Two prominent algorithms for solving RL tasks, which we used in our work, are the value-based DQN [Mnih et al., 2015] and the policy-based PPO [Schulman et al., 2017].

Deep Q Networks (DQN): The DQN algorithm approximates the optimal Q function with a Convolutional Neural Network (CNN), by optimizing the network weights such that the expected Temporal Difference (TD) error of the optimal 'bellman Equation is minimized:

$$||r(s_t) + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a'; \theta_{\text{target}}) - Q(s_t, a_t; \theta)||_2^2 \,.$$

The DQN is an off-policy learning algorithm, i.e., transitions are stored in the Experience Replay and used for training. When applying minibatch training updates, the DQN samples tuples of experience at random from the pool of stored samples in the buffer. In addition, the DQN maintains two separate Q-networks: the current Q-network with parameters θ, and a target Q-network with parameters θ_{target}. Finally, every T_{update} (fixed number of iterations) the target network is updated to be equal to the current one.

Proximal Policy Optimization (PPO): In contrast to the DQN, PPO is an on-policy algorithm. PPO uses two estimators, one for the policy π, and one for the value v^π. Training is performed by interleaving data collection and policy optimization. Once a batch of new transitions is collected, PPO attempts to maximize the objective

$$\min\left(\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} \hat{A}^{\text{GAE}}, \text{clip}\left(\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)}, 1-\epsilon, 1+\epsilon\right) A^{\text{GAE}}\right),$$

where $\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)}$ is the ratio of the probability of taking the given action under the current policy π to that of taking the same action under the previous policy π_{old} used to collect the data. ϵ is a hyperparameter which controls the amount of clipping. Intuitively, the loss encourages the policy to improve its behavior whereas the clipping prevents it from deviating, thereby forcing conservative updates.

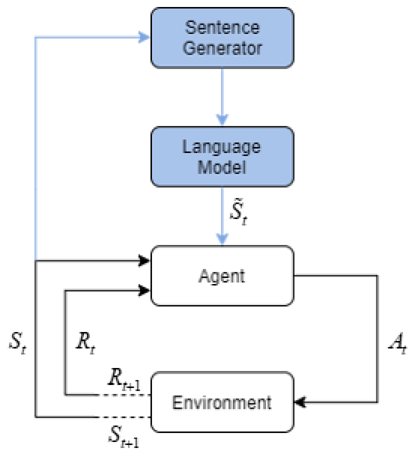

Figure 1: Block diagram for using language-based state representation in RL.

3 Semantic Representation Methods

Contemporary methods for semantic representation of states follow one of three approaches: (1) raw visual inputs [Mnih et al., 2015; Kempka et al., 2016], in which raw sensory values of pixels are used from one or multiple sources, (2) feature vectors [Todorov et al., 2012], in which general features of the problem are chosen, with no specific structure, and (3) semantic segmentation maps [Ronneberger et al., 2015], in which discrete or logical values are used in one or many channels to represent the general features of the state.

In RL, the raw form often corresponds to pixels representing an image. However, the image is only one form of a semantic representation. In semantic segmentation, the image is converted from a 3-channel (RGB) matrix into an N-channel matrix, where N is the number of classes. In this case, each channel represents a class, and a binary value at each coordinate denotes whether or not this class is present in the image at this location. Semantic segmentation maps contain less **task-nuisances** [Achille and Soatto, 2018] than raw images. Task nuisances are random variables that affect the observed data, but are not informative to the task we are trying to solve.

In this paper we propose an additional method for representing a state, namely using natural language descriptions. One method to achieve such a representation is through image captioning [Hossain et al., 2019]. Natural language is both rich as well as flexible. This flexibility enables the algorithm designer to represent the information present in the state as efficiently and compactly as possible. In addition, natural language can contribute to establishing interpretable agents, a key component of Explainable Artificial Intelligence [Samek et al., 2017].

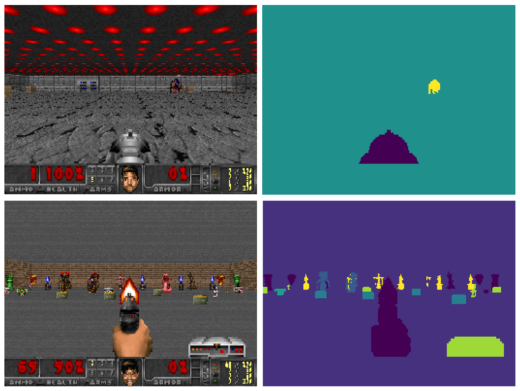

Figure 2: Raw visual inputs and their corresponding semantic segmentation in the VizDoom enviornment.

3.1 Natural Language State Representation

In this work, we represent the state using natural language, i.e., *words*. One approach for representing natural language sentences is through word embeddings. A word embedding is a mapping from a word w to a vector $\mathbf{w} \in \mathbb{R}^d$. A simple form of word embedding is the Bag of Words (BoW) – a vector $\mathbf{w} \in \mathbb{N}^{|D|}$ ($|D|$ is the dictionary size), in which each word receives a unique 1-hot vector representation. Recently, more efficient methods have been proposed, in which the embedding vector is smaller than the dictionary size, $d \ll |D|$. These methods are also known as distributional embeddings.

The distributional hypothesis in linguistics is derived from the semantic theory of language usage (i.e., words that are used and occur in the same contexts tend to have similar meanings). Distributional word representations are a fundamental building block for representing natural language sentences. Word embeddings such as Word2vec [Mikolov et al., 2013] and GloVe [Pennington et al., 2014] build upon the distributional hypothesis, improving efficiency of state-of-the-art language models.

While the common theme in NLP is to use recurrent neural networks, Convolutional Neural Networks (CNNs), originally invented for computer vision, have been shown to achieve strong performance on text classification tasks as well as other traditional NLP tasks [Collobert et al., 2011]. In this paper we consider a common architecture, the Text-CNN [Kim, 2014], which has been used extensively in NLP domains and enjoys lower complexity and is thus a natural candidate for RL tasks [Tessler et al., 2019].

4 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. We therefore expect these representations to obtain superior performance for solving tasks in ViZDoom. Moreover, as we will see in Section 5, a natural language augmentation of these representations can achieve higher performance and robustness.

4.1 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.

Limiting Information

In real world applications, natural language can be efficiently used to describe non-spatial attributes that would be hard otherwise. A designer can inject natural language descriptions with subjective or time-dependent information. 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. Furthermore, to allow for flexibility in the complexity of the state, the number of patches was varied in controlled experiments, as discussed in Section 5.3.

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. We introduced ambiguity in our parser through a mixture of distinct parsers.

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

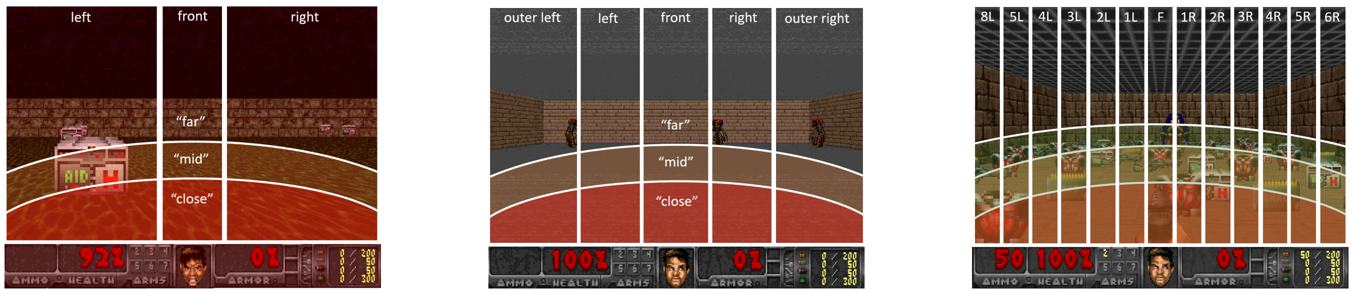

Figure 3: Frame division used for describing the state in natural language.

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.

5 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. Frequently, the agent may suffer from constant health degeneration, for instance, due to poison. The different scenarios include a scenario in which the agent must take cover from inbound fireballs (Defend the Center), a scenario in which the agent must defend itself from charging enemies (Defend the Line), 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 as described in Section 3.1. 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 visual representation techniques; namely, the raw image and semantic segmentation maps.

In order to effectively compare the performance of the different representation methods, we conducted our experiments under similar conditions. First, the same hyper-parameters were used for all representations. Second, to rule out effects of architectural expressiveness, we validated performance over different number of weights and used those that provided best results for visual-based representations. Finally, we ensured the "super" scenario was positively biased toward image-based representations. This was performed by adding a large amount of items to the game level, thereby filling the state with nuisances. This is especially evident in the NLP representations, as the complexity of learning an optimal policy grows with the sentence's length. This is contrary to image-based representations, in which the input dimension remained constant. We observed an average of over 250 words-per-state.

5.1 Performance

Results for DQN and PPO-based agents are presented in Fig. 4. Each plot depicts the average reward (across 5 seeds) of all representations 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. This difference decreases as the complexity of the scenarios increases. Still, NLP representations maintain consistent and robust results throughout all tested scenarios.

A more thorough analysis of the performance shows that although natural language eventually outperforms other representations, the difference and convergence time are domain dependent. For example, in the Defend the Line scenario, NLP representations did not have a clear advantage over semantic segmentation maps with the DQN agent.

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. We note that this is not related to characteristics of the architecture, but rather to the semantic representation technique and the information required to solve the task.

5.2 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. Moreover, since NLP representations were confronted with ambiguity through random parsers, they were effectively influenced by multiple sources of noise throughout training.

The robustness of NLP representations to task-nuisances indicate their ability to summarize information well. Particularly, the ability to count the elements in each patch mitigated the nuisance effect. We note that, while the raw visual representations were least affected by nuisance, their overall performance remained much smaller than the other representation techniques.

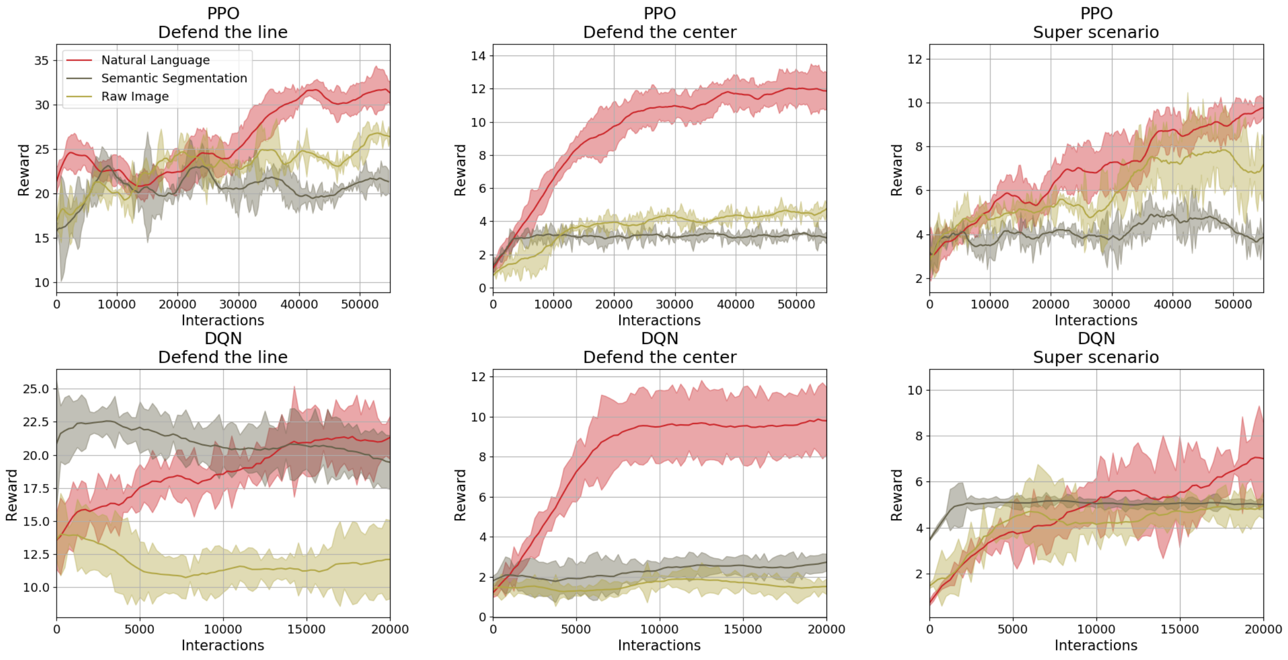

Figure 4: Comparison of representation methods on the different VizDoom scenarios using DQN and PPO agents. X and Y axes represent the number of iterations and cumulative reward, respectively.

5.3 Discretization to Patches

Discretization is a commonly used method in RL for lowering the dimensionality of the state space or action space. It allows to efficiently solve problems with a large state space or continuous action space. This approach may come at the expense of finding a suboptimal solution. Nevertheless, adaptive discretization techniques may be used to obtain better overall performance [Sinclair et al., 2019].

The 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.

These experiments were conducted over the ”super” scenario to ensure a high probability of non-empty patches. The extreme case of 31 patches rendered sentence lengths of over 400 words with no evident loss in performance, indicating that sentence length did not have an effect on performance. Moreover, these results suggest that discretization in the ViZDoom environment does not affect performance.

6 Related Work

Work on representation learning is concerned with finding an appropriate representation of data in order to perform a machine learning task [Goodfellow et al., 2016]. In particular, deep learning exploits this concept by its very nature [Mnih et al., 2015]. Work on representation learning include Predictive State Representations (PSR) [Littman and Sutton, 2002], which capture the state as a vector of predictions of future outcomes, and a Heuristic Embedding of Markov Processes (HEMP) [Engel and Mannor, 2001], which learns to embed transition probabilities using an energy-based optimization problem.

There has been extensive work attempting to use natural language in RL. Efforts that integrate language in RL develop tools, approaches, and insights that are valuable for improving the generalization and sample efficiency of learning agents. Previous work on language-conditioned RL has considered the use of natural language in the observation and action space. Environments such as Zork and TextWorld [Côté et al., 2018; Tessler et al., 2019] have been the standard benchmarks for testing text-based games. Nevertheless, these environments do not search for semantic state representations, in which an RL algorithm can be better evaluated and controlled.

More recently, the structure and compositionality of natural language has been used for representing policies in hierarchical RL. In a paper by Hu et al. [2019], instructions given in natural language were used in order to break down complex problems into high-level plans and lower-level actions. Their suggested framework leverages the structure inherent to natural language, allowing for transfer to unfamiliar tasks and situations. This use of semantic structure has also been leveraged by Tennenholtz and Mannor [2019], where abstract actions (not necessarily words) were recognized as symbols of a natural and expressive language, improving performance and transfer of RL agents.

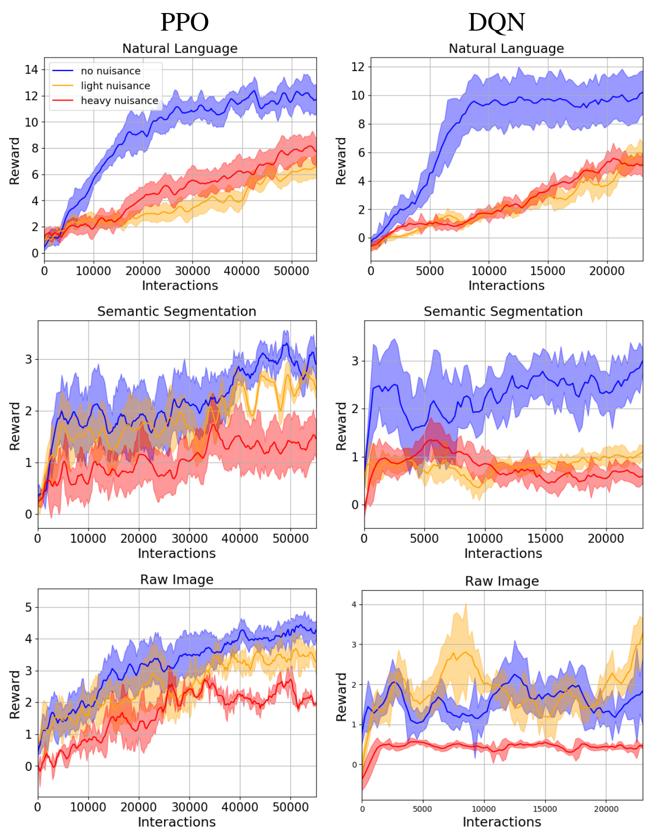

Figure 5: Robustness of each representation type with respect to amount of nuisance, tested on "Defend the Center" scenario.

7 Discussion and Future Work

Our results indicate that natural language can outperform, and sometime even replace, vision-based representations. Nevertheless, natural language representations can also have disadvantages in various scenarios. For one, they require the designer to be able to describe the state exactly, whether by a rule-based or learned parser. Second, they abstract notions of the state space that the designer may not realize are necessary for solving the problem. As such, semantic representations should be carefully chosen, similar to the process of reward shaping or choosing a training algorithm. Here, we enumerate three instances in which we believe natural language representations are beneficial:

Natural use-case: Information contained in both generic and task-specific textual corpora may be highly valuable for decision making. This case assumes the state can either be easily described using natural language or is already in a natural language state. This includes examples such as user-based domains, in which user profiles and comments are part of the state, and the stock market, in which stocks are described by analysts and other readily available text. 3D physical environments such as VizDoom also fall into this category, as semantic segmentation maps can be easily described using natural language.

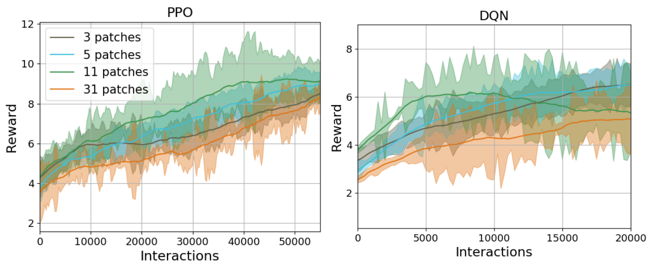

Figure 6: Average rewards of NLP based agent as a function of the number of patches in the language model.

Subjective information: Subjectivity refers to aspects used to express opinions, evaluations, and speculations. These may include strategies for a game, the way a doctor feels about her patient, the mood of a driver, and more. While our work did not focus on subjective information, it is a form of information that is hard to represent in non-textual forms, including images or feature vectors.
Unstructured information: In these cases, features might be measured by different units, with an arbitrary position in the state's feature vector, rendering them sensitive to permutations. Such state representations are difficult to process using neural networks. As an example, the medical domain may contain numerous features describing the vitals of a patient. These raw features, when observed by an expert, can be efficiently described using natural language. Moreover, they allow an expert to efficiently add subjective information.

An orthogonal line of research considers automating the process of image annotation. While this work considered a manually constructed language parser, general language generators can be used to automatically extract meaningful textual explanations. In addition, while we have only considered spatial features of the state, subjective and transient information can be efficiently encoded as well.

An additional future direction focuses on agent accountability, a pillar of Explainable Artificial Intelligence [Samek et al., 2017]. Transforming states into natural language can be used as a tool to discern competence. An RL system that is accountable can explain why it acts in a certain way. The explanation can be reasoning in words, by example, or in any other means interpretable to the other entities that communicate with it. It is thus valuable to design interpertable agents that can leverage language in states, actions, and time.

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

Omri Abend and Ari Rappoport. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

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

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537, 2011.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. *arXiv preprint arXiv:1806.11532*, 2018.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Yaakov Engel and Shie Mannor. Learning embedded maps of markov processes. In *in Proceedings of ICML 2001*. Citeseer, 2001.

John R Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, 1957.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

MD Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 2019.

Hengyuan Hu, Denis Yarats, Qucheng Gong, Yuandong Tian, and Mike Lewis. Hierarchical decision making by generating and following natural language instructions. In *Advances in neural information processing systems*, 2019.

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.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *International conference on machine learning*, 2016.

Michael L Littman and Richard S Sutton. Predictive representations of state. In *Advances in neural information processing systems*, pages 1555–1561, 2002.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

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.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, 2016.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.

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

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

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.

Sean R Sinclair, Siddhartha Banerjee, and Christina Lee Yu. Adaptive discretization for episodic reinforcement learning in metric spaces. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(3):1–44, 2019.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Guy Tennenholtz and Shie Mannor. The natural language of actions. In *International Conference on Machine Learning*, pages 6196–6205, 2019.

Chen Tessler, Tom Zahavy, Deborah Cohen, Daniel J Mankowitz, and Shie Mannor. Action assembly: Sparse imitation learning for text based games with combinatorial action spaces. *arXiv preprint arXiv:1905.09700*, 2019.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

Adina Williams, Andrew Drozdov, and Samuel R Bowman. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association for Computational Linguistics*, 6:253–267, 2018.