Reading and Acting while Blindfolded: The Need for Semantics in Text Game Agents

Shunyu Yao*
Princeton University
shunyuy@princeton.edu

Karthik Narasimhan
Princeton University
karthikn@princeton.edu

Matthew Hausknecht
Microsoft Research
matthew.hausknecht@microsoft.com

Abstract

Text-based games simulate worlds and interact with players using natural language. Recent work has used these games as a testbed for autonomous language-understanding agents, with the motivation being that understanding the *semantics* or meanings of words is a key component of how humans understand, reason, and act in these worlds. However, it remains unclear to what extent artificial agents utilize semantic understanding of the text. To this end, we perform experiments to degrade the amount of semantic information available to a learning agent. Surprisingly, we find that an agent is capable of preserving similar scores even in the complete absence of language semantics, indicating current agents may be poorly designed to understand and leverage game texts. To remedy this deficiency, we propose an inverse dynamics decoder which helps regularize the representation space, encourages exploration, and shows improved performance on several games including ZORK I. We discuss the implications of our findings for designing future agents with stronger semantic understanding.

1 Introduction

Text adventure games such as ZORK I (Figure 1 (a)) have been a testbed for developing autonomous agents that operate using natural language. Since interactions in these games (input observations, action commands) are through text, the ability to understand and use language is deemed necessary and critical to progress through such games. Previous work has deployed a spectrum of methods for language processing in this domain, including word vectors [5], recurrent neural networks [11, 7], pre-trained language models [13], open-domain question answering systems [3], knowledge graphs [2, 3, 1], and reading comprehension models [6].

Meanwhile, most of these models operate under the reinforcement learning (RL) framework, where the agent explores the same environment in repeated episodes, learning a value function or policy to maximize game score. From this perspective, text games are just special cases of a partially observable Markov decision process (POMDP) (S, T, A, O, R, γ), where players issue text actions $a \in A$, receive text observations $o \in O$ and scalar rewards $r = R(s, a)$, and the underlying game state $s \in S$ is updated by transition $s' = T(s, a)$.

However, what distinguishes these games from other POMDPs is the fact that the actions and observations are in language space L. Therefore, a certain level of decipherable *semantics* is attached

*Work partly done during internship at Microsoft.

Wordplay: When Language Meets Games Workshop @ NeurIPS 2020,
https://wordplay-workshop.github.io/

(a) ZORK I

Observation 21: You are in the living room. There is a doorway to the east, a wooden door with strange gothic lettering to the west, which appears to be nailed shut, a trophy case, and a large oriental rug in the center of the room. You are carrying: A brass lantern ...

Action 21: move rug

Observation 22: With a great effort, the rug is moved to one side of the room, revealing the dusty cover of a closed trap door... Living room... You are carrying: ...

Action 22: open trap

(b) **MIN-OB**

Observation 21: Living Room
Action 21: move rug
Observation 22: Living Room
Action 22: open trap

(c) **HASH**

Observation 21: 0x6fc
Action 21: 0x3a04
Observation 22: 0x103b
Action 22: 0x16bb

Figure 1: (a): Sample original gameplay from ZORK I. (b) (c): Two proposed semantics ablations. (b) **MIN-OB** reduces observations to only the current location name, and (c) **HASH** replaces observation and action texts by their string hash values.

to text observations $o \in O \subset L$ and actions $a \in A \subset L$. Ideally, these texts not only serve as observation or action *identifiers*, but also provide clues about the latent transition function T and reward function R. For example, issuing "jump" from observation "on the cliff" would likely yield a subsequent observation like "you are killed" along with a negative reward. Human players often rely on their understanding of language and its semantics to inform their choices, while replacing texts with non-semantic identifiers such as their corresponding hashes (Figure 1 (c)) would likely render games unplayable for people. However, would this type of transformation affect current RL agents developed for such games? In this work, we ask the following question: *To what extent do current reinforcement learning agents leverage semantics in text-based games?*

To shed light on this question, we investigate the Deep Reinforcement Relevance Network (DRRN) [10], a top-performing RL model that uses gated recurrent units (GRU) [4] to encode texts. We conduct three experiments on a set of games from the Jericho benchmark [7] to probe the effect of different semantic representations on the functioning of DRRN. These include (1) using just a location phrase as the input observation (Figure 1 (b)), (2) hashing text observations and actions (Figure 1 (c)), and (3) regularizing vector representations using an auxiliary inverse dynamics loss. While reducing observations leads to decreased scores and enforcing inverse dynamics decoding leads to increased scores on some games, hashing texts to break semantics surprisingly matches or even outperforms the baseline DRRN on almost all games considered. This implies current RL agents for text-based games might not be sufficiently leveraging the semantic structure of game texts to learn good policies, and points to the need for developing agents that have a better grasp of natural language.

2 Models

DRRN Baseline Our baseline RL agent DRRN [9] learns a Q-network $Q_\phi(o, a)$ parametrized by ϕ. The model encodes the observation o and each action candidate a using two separate GRU encoders f_o and f_a, and then aggregates the representations to derive the Q-value through a MLP decoder g:

$$Q_\phi(o, a) = g(\text{concat}(f_o(o), f_a(a))) \tag{1}$$

For learning ϕ, tuples (o, a, r, o') of observation, action, reward and the next observation are sampled from an experience replay buffer and the following temporal difference (TD) loss is minimized:

$$\mathcal{L}_{\text{TD}}(\phi) = (r + \gamma \max_{a' \in A} Q_\phi(o', a') - Q_\phi(o, a))^2 \tag{2}$$

During gameplay, a softmax exploration policy is used to sample an action:

$$\pi_\phi(a|o) = \frac{\exp(Q_\phi(o, a))}{\sum_{a' \in A} \exp(Q_\phi(o, a'))} \tag{3}$$

Note that when the action space A is large, (2) and (3) become intractable. A valid action handicap [7] or a language model [13] can be used to generate a reduced action space for efficient exploration. For all the modifications below, we use the DRRN with the valid action handicap as our base model.

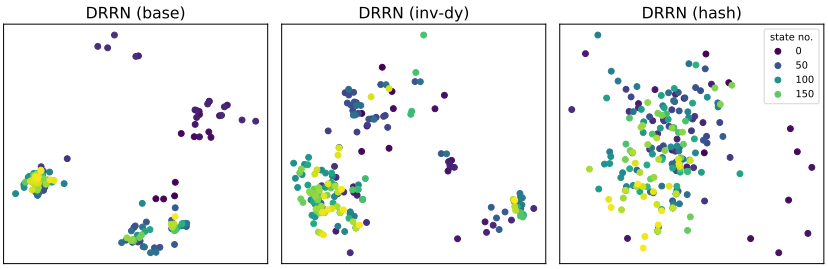

Figure 2: PCA visualization of the first 200 walkthrough state observations of ZORK I.

Reducing Semantics via Minimizing Observation (MIN-OB) Unlike other RL domains such as video games or robotics control, at each step of text games the (valid) action space is constantly changing, and it reveals useful information about the current state. For example, knowing "unlock box" is valid leaks the existence of a locked box. Also, sometimes action semantics indicate its value even unconditional on the state, e.g. "pick gold" usually seems good. Given these, we minimize the observation to only a location phrase $o \mapsto \text{loc}(o)$ (Figure 1 (b)) to isolate the action semantics: $Q_\phi^{\text{loc}}(o, a) = g(f_o(\text{loc}(o)), f_a(a)))$.

Breaking Semantics via Hashing (HASH) GRU encoders f_o and f_a in the Q-network (1) generally ensure that similar texts (e.g. a single word change) are given similar representations, and therefore similar values. To study whether such a semantics continuity is useful, we break it by hashing observation and action texts. Specifically, given a hash function from strings to integers $h : L \to \mathbb{Z}$, and a pseudo-random generator $G : \mathbb{Z} \to \mathbb{R}^d$ that turns an integer seed to a random Gaussian vector, a hashing encoder $\hat{f} = G \circ h : L \to \mathbb{R}^d$ can be composed. While f_o and f_a in (1) are trainable, $\hat{f}$ is fixed throughout RL, and ensures two texts that only differ a word would have completely different representations. In this sense, hashing breaks semantics and only serves to identify different observations and actions in an abstract MDP problem (Figure 1 (c)): $Q_\phi^{\text{hash}}(o, a) = g(\hat{f}(o), \hat{f}(a))$.

Regularizing Semantics via Inverse Dynamics Decoding (INV-DY) The GRU representations in DRRN $f_o(o), f_a(a)$ are only optimized for the TD loss (2). As a result, text semantics can degenerate during encoding, and the text representation might arbitrarily overfit to the Q-values. To regularize and encourage more game-related semantics to be encoded, we take inspiration from Pathak et al. [12] and propose an inverse dynamics auxiliary task during RL. Given representations of current and next observations $f_o(o), f_o(o')$, we use a MLP g_{inv} to predict the action representation, and a GRU decoder d to decode the action back to text[†]. The inverse dynamics loss is defined as

$$\mathcal{L}_{\text{inv}}(\phi, \theta) = -\log p_d(a | g_{inv}(\text{concat}(f_o(o), f_o(o')))) \tag{4}$$

where θ denote weights of g_{inv} and d, and $p_d(a|x)$ is the probability of decoding token sequence a using GRU decoder d with initial hidden state as x. To also regularize the action encoding, action reconstruction from f_a is also used as a loss term:

$$\mathcal{L}_{\text{dec}}(\phi, \theta) = -\log p_d(a | f_a(a)) \tag{5}$$

And during experience replay, these two losses are optimized along with the TD loss:

$$\mathcal{L}(\phi, \theta) = \mathcal{L}_{\text{TD}}(\phi) + \lambda_{inv}\mathcal{L}_{\text{inv}}(\phi, \theta) + \lambda_{dec}\mathcal{L}_{\text{dec}}(\phi, \theta) \tag{6}$$

An intrinsic reward $r^+ = \mathcal{L}_{\text{inv}}(\phi, \theta)$ is also used to explore toward where the inverse dynamics is not learned well yet. All in all, the aim of **INV-DY** is threefold: (1) regularize both action and observation representations to avoid degeneration by decoding back to the textual domain, (2) encourage f_o to encode action-relevant parts of observations, and (3) provide intrinsic motivation for exploration.

3 Results

Setup We train on 12 games from the Jericho benchmark [7]. For each game, we train DRRN asynchronously on 8 parallel instances of the game environment for 10^5 steps, using a prioritized

[†]Directly defining an L1/L2 loss between $g_{inv}(\text{concat}(f_o(o), f_o(o')))$ and $f_a(a)$ will collapse text representations together.

Game	DRRN	MIN-OB	HASH	INV-DY	Max
balances	10 / 10	10 / 10	10 / 10	10 / 10	51
deephome	57 / 66	8.5 / 27	58 / 67	57.6 / 67	300
detective	290 / 336.7	86.3 / 350	290 / 316.7	290 / 323.3	360
dragon	-5.0 / 6	-5.4 / 3	-5.0 / 7	**-2.7** / **8**	25
enchanter	20 / 20	20 / 40	20 / 30	20 / 30	400
inhumane	21.1 / 45	12.4 / 40	21.9 / 45	19.6 / 45	90
library	15.7 / 21	12.8 / 21	**17** / 21	16.2 / 21	30
ludicorp	12.7 / 23	11.6 / 21	**14.8** / 23	13.5 / 23	150
omniquest	4.9 / 5	4.9 / 5	4.9 / 5	5.3 / **10**	50
pentari	26.5 / 45	21.7 / 45	**51.9** / **60**	37.2 / 50	70
zork1	39.4 / 53.3	29 / 46	35.5 / 50	**43.1** / **87**	350
zork3	0.4 / 4.5	0.0 / 4	0.4 / 4	0.4 / 4	7
Avg. Norm	.21 / .38	.12 / .35	**.25** / .39	.23 / **.40**	

Table 1: Final score / maximum score of different models.

replay buffer. Following prior practice [7], we augment observations with location and inventory descriptions by issuing the 'look' and 'inventory' commands. We train three independent runs for each game and report their average score. For **HASH**, we use the Python built-in hash function to process text as a tuple of token IDs, and implement the random vector generator G by seeding PyTorch with the hash value. For **INV-DY**, we use $\lambda_{inv} = \lambda_{dec} = 1$.

Scores Table 1 reports the final score (the average score of the final 100 episodes during training), and the maximum score seen in each game for different models. Average normalized score (raw score divided by game total score) over all games is also reported. Compared to the base DRRN, **MIN-OB** turns out to explore similar maximum scores on most games (except DEEPHOME and DRAGON), but fails to memorize the good experience and reach high episodic scores, which suggests the importance of identifying different observations using language details. Most surprisingly, **HASH** has a lower final score than DRRN on only one game (ZORK I), while on PENTARI it almost doubles the DRRN final score. It is also the model with the best average normalized final score across games, which indicates that the DRRN model can perform as well without leveraging any language semantics, but instead simply by identifying different observations and actions with random vectors and memorizing the Q-values. Lastly, we observe on some games (DRAGON, OMNIQUEST, ZORK I) **INV-DY** can explore high scores that other models cannot. Most notably, on ZORK I the maximum score seen is 87 (average of $54, 94, 113$), while any run of other models does not explore a score more than 55. This might indicate potential benefit of developing RL agents with more semantic representations.

Visualizations In Figure 2, we use PCA to visualize observation representations for the first 200 walkthrough states of ZORK I. For DRRN and **INV-DY**, observations within few steps often have similar representations, while **HASH** has no such property and representations are randomly scattered. In addition, **INV-DY** represents unseen observations (state no. >100) with more variations than DRRN, and their representations mix with experienced observations as a result of their shared aspects of semantics. This helps explain the better exploration of **INV-DY** on this game. On the other hand, DRRN overfits to the TD loss (2) alone, thus represents unseen observations arbitrarily collapsed and far from seen observations, which might hinder gameplay at the future stage.

4 Discussion

At a high level, RL agents for text-based games succeed by (1) exploring trajectories that lead to high scores, and (2) learning representations to stably reach high scores. From our experiments, we find that a semantics-regularized **INV-DY** model manages to explore higher scores on some games (DRAGON, OMNIQUEST, ZORK I), while the **HASH** model manages to memorize scores better on other games (LIBRARY, LUDICORP, PENTARI) using just a fixed, random, non-semantic representation. This leads us to hypothesize two things. First, fixed, stable representations might make downstream Q-learning easier. Second, it might be desirable to represent similar texts very differently for better gameplay, e.g. the Q-value can be much higher when a key object is mentioned, even if it only adds a few words to a long observation text. This motivates future thought into the structural vs. functional use of language semantics in these games.

Our findings also urge a re-thinking of the popular 'RL + valid action handicap' setup for these games. RL in a text environment with limited corpora and sparse rewards may lead to overfitting (Figure 2), and the valid action handicap may lift away too much of the language understanding challenge for the RL agent. We present the **HASH** model as a strong baseline with no proper semantics under this scheme, and its strong relative performance indicates that the handicap may be alleviating a considerable amount of language understanding challenges for RL agents. This also advocates for more attention towards the 'no-handicap' setting [13, 8], where *generating* action candidates rather than simply *choosing* from a set entails more opportunities and challenges with respect to language semantics.

References

[1] A. Adhikari, X. Yuan, M.-A. Côté, M. Zelinka, M.-A. Rondeau, R. Laroche, P. Poupart, J. Tang, A. Trischler, and W. L. Hamilton. Learning dynamic belief graphs to generalize on text-based games, 2020.

[2] P. Ammanabrolu and M. Hausknecht. Graph constrained reinforcement learning for natural language action spaces. *arXiv preprint arXiv:2001.08837*, 2020.

[3] P. Ammanabrolu, E. Tien, Z. Luo, and M. O. Riedl. How to avoid being eaten by a grue: Exploration strategies for text-adventure agents. *arXiv preprint arXiv:2002.08795*, 2020.

[4] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[5] N. Fulda, D. Ricks, B. Murdoch, and D. Wingate. What can you do with a rock? affordance extraction via word embeddings. *CoRR*, abs/1703.03429, 2017.

[6] X. Guo, M. Yu, Y. Gao, C. Gan, M. Campbell, and S. Chang. Interactive fiction game playing as multi-paragraph reading comprehension with reinforcement learning, 2020.

[7] M. Hausknecht, P. Ammanabrolu, M.-A. Côté, and X. Yuan. Interactive fiction games: A colossal adventure. *CoRR*, abs/1909.05398, 2019.

[8] M. Hausknecht, R. Loynd, G. Yang, A. Swaminathan, and J. D. Williams. Nail: A general interactive fiction agent. *arXiv preprint arXiv:1902.04259*, 2019.

[9] J. He, J. Chen, X. He, J. Gao, L. Li, L. Deng, and M. Ostendorf. Deep reinforcement learning with a natural language action space. *arXiv preprint arXiv:1511.04636*, 2015.

[10] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Austin, Texas, Nov. 2016. Association for Computational Linguistics.

[11] K. Narasimhan, T. D. Kulkarni, and R. Barzilay. Language understanding for text-based games using deep reinforcement learning. In *EMNLP*, pages 1–11, 2015.

[12] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

[13] S. Yao, R. Rao, M. Hausknecht, and K. Narasimhan. Keep calm and explore: Language models for action generation in text-based games. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2020.