Modeling Worlds in Text

Anonymous Author(s)
Affiliation Address
email

Abstract

We provide a dataset that enables the creation of learning agents that can build knowledge graph-based world models of interactive narratives.[1] Interactive narratives—or text-adventure games—are partially observable environments structured as long puzzles or quests in which an agent perceives and interacts with the world purely through textual natural language. Each individual game typically contains hundreds of locations, characters, and objects—each with their own unique descriptions—providing an opportunity to study the problem of giving language-based agents the structured memory necessary to operate in such worlds. Our dataset provides 24198 mappings between rich natural language observations and: (1) knowledge graphs that reflect the world state in the form of a map; (2) natural language actions that are guaranteed to cause a change in that particular world state. The training data is collected across 27 games in multiple genres and contains a further 7836 heldout instances over 9 additional games in the test set. We further provide baseline models using rules-based, question-answering, and sequence learning approaches in addition to an analysis of the data and corresponding learning tasks.

1 Introduction

We seek to create agents that exhibit human-like capabilities such as commonsense reasoning and natural language understanding in interactive and situated settings. Interactive narrative environments provide a critical stepping stone in this pursuit towards creating learning agents that can produce contextually relevant and goal-driven natural language [Côté et al., 2018, Urbanek et al., 2019, Hausknecht et al., 2020]. They require agents to observe textual descriptions and then act upon the world using natural language with the aim of completing a long term goal or quest as seen in Figures 1, 2. We focus on two of the core challenges faced by learning agents in these environments—as identified in prior work—knowledge representation and a combinatorially sized state-action space.

Observation: **West of House** You are standing in an open field west of a white house, with a boarded front door. There is a small mailbox here.

Action: **Open mailbox**

Observation: Opening the small mailbox reveals a leaflet.

Action: **Read leaflet**

Observation: (Taken) "WELCOME TO ZORK! ZORK is a game of adventure, danger, and low cunning. In it you will explore some of the most amazing territory ever seen by mortals. No computer should be without one!"

Action: **Go north**

Observation: **North of House** You are facing the north side of a white house. There is no door here, and all the windows are boarded up. To the north a narrow path winds through the trees.

Figure 1: Excerpt from *Zork1*.

The **knowledge representation** challenge rises from the fact that interactive narratives span many distinct locations, each with unique descriptions, objects, and characters as seen can be seen in Figure 2. Players move by issuing navigational

[1]Dataset can be found here https://github.com/JerichoWorld/JerichoWorld

Figure 2: A map showcasing the size and complexity of the world of *Zork* by artist *ion_bond*.

commands, which can convey Euclidean space like *go West* or non-Euclidean span like *step into portal*, warping the agent to an entirely new section of the world. To cope with such challenges, humans often create structured memory aids such as hand drawn maps when attempting to play these games. A good knowledge representation can assist with long-term action dependencies that often arise in game quests (as well as real world environments). An example of a long-term dependency is a key being found in one location that opens a lock on a chest in an entirely different section of the map. For an agent to learn this relationship, it must be able to replicate the sequence of picking up the key and unlocking the chest while not being distracted by interstitial actions and states.

Long-term action dependencies are made challenging by two aspects of interactive narrative environments, which are also present in real-world environments. First, these environments are **partially observable** in the sense that an agent only has local observability. Second, interactive narrative environments have a **combinatorially-sized natural language state-action space**. For example, in the cannonical game *Zork1* an action can consist of up to five-words from a relatively modest vocabulary of 697 words, resulting in $\mathcal{O}(697^5) = 1.64 \times 10^{14}$ possible actions at every step—though the number of *valid actions* that are gramatically coherent and contextually relevant is significantly smaller. This makes exploration sample-inefficient, making it harder to learn the relationship between actions that are temporally distant from each other.

The knowledge representation challenges inherent to interactive narrative games give rise to the **Textual-SLAM** problem, a textual variant of Simultaneous Localization And Mapping (SLAM) [Thrun et al., 2005] problem of constructing a map by *inferring information* from one's surroundings while navigating a novel environment. As in humans, the creation of such world models or memory aids in agents—in the form of knowledge graphs—has been shown to be critical in helping automated learning agents operate in these textual worlds [Ammanabrolu and Riedl, 2019, Murugesan et al., 2020, Adhikari et al., 2020, Ammanabrolu and Hausknecht, 2020].

Despite the success of knowledge graphs in addressing these problems, a broad dataset across a diverse set of games mapping text game observations to knowledge graphs does not exist—hindering progress in building of world modeling agents with structured memory. Building off the popular text game simulator Jericho [Hausknecht et al., 2020], we have constructed a dataset dubbed JerichoWorld that maps text game state observations to both the underlying ground truth knowledge graph representations of the game and the set of contextually relevant actions that can be performed in that state. Using this data, we seek to enable development of agents that focus on answering the questions of "What actions make sense for me to perform right now?" and "What have I already done and how will the world change now if I perform a particular action?"—questions relating to the problems natural language understanding, commonsense reasoning, and structured memory. The training set contains 24198 instances across 27 games and the heldout test set contains 7836 instances from 9 games. We further formally define two initial tasks for this dataset focusing on the questions mentioned: (1) Given a textual observation, predict the underlying knowledge graph of the world. (2) Given a textual observation, predict the set of actions that are contextually relevant. Results for three baselines—using rules-based, question-answering, and sequence-learning approaches—are provided in addition to an analysis of the dataset and results themselves.

2 Related Work

We constrain our related work section to three primary areas: current interactive narrative benchmarks, world modeling and model-based reinforcement learning, and the use of knowledge graphs in text games. Currently, three primary open-source platforms and baseline benchmarks have been developed so far to help measure progress in this field: *Jericho* [Hausknecht et al., 2020][2] a learning environment for human-made interactive narrative games; *TextWorld* [Côté et al., 2018][3] a framework for procedural generation in text-games; and *LIGHT* [Urbanek et al., 2019][4] a large-scale crowdsourced multi-user text-game for studying situated dialogue. Further extensions and adaptation to some of these benchmarks have been proposed for use in neighboring domains such as vision-and-language navigation [Shridhar et al., 2021], commonsense reasoning [Murugesan et al., 2021], and procedural text understanding [Tamari et al., 2021]. Our work builds on the Jericho environment.

Work on world models in learning agents have recently been inspired by theories of how humans form mental models of the world [Jancke, 2000, Ha and Schmidhuber, 2018]. When in the form of predictive probabilistic generative models of the world, they can be used in model-based reinforcement learning tasks [Sutton and Barto, 1998, Arulkumaran et al., 2017, Schrittwieser et al., 2019]. In such cases, a learning agent attempts to learn the underlying environment dynamics at the same time as a policy, often using information about one to inform the other. Ha and Schmidhuber [2018] take this one step further by replacing a environment entirely with the agent's own learned world model and training a control policy there. All of these methods have been shown to have the added benefits of significantly improving sample efficiency as the agent is now able to (at least partially) simulate the environment via the world model.

In all of the world modeling cases mentioned, the state representations that the models are conditioned on are drawn directly from the existing base environments, e.g. raw pixel game screens in the case of the Arcade Learning Environment [Bellemare et al., 2013] or other visual games such as Sokoban [Bamford and Lucas, 2020]. In the case of human-made text games, however, knowledge graphs—not directly provided by existing text game learning frameworks—have been shown to be superior state representations when compared to just the textual observations by themselves. They aid in the challenges of partial observability/knowledge representation [Ammanabrolu and Riedl, 2019, Adhikari et al., 2020, Sautier et al., 2020], combinatorial state-action spaces [Ammanabrolu and Hausknecht, 2020, Ammanabrolu et al., 2020b], and commonsense reasoning [Ammanabrolu and Riedl, 2019, Murugesan et al., 2020, 2021, Dambekodi et al., 2020].

Closest in spirit to this work is the Jericho-QA dataset [Ammanabrolu et al., 2020b], a question-answering dataset tuned to text games that enables agents to identify common objects in the world and their attributes. It does not have information regarding the full underlying knowledge graph state or valid actions, however. As far as we know, ground truth knowledge graph state representation dataset across a diverse set of human-made text games is not currently available in any of the primary text game benchmarks mentioned previously, hindering the ability to create agents with structured memory in the form of graph-based world models.

3 JerichoWorld

Côté et al. [2018] and Hausknecht et al. [2020] define text games as Partially-Observable Markov Decision Processes. A game can be represented as a 7-tuple of $\langle S, T, A, \Omega, O, R, \gamma \rangle$ representing the set of environment states, *mostly deterministic conditional transition probabilities between states*, the vocabulary or words used to compose text commands, observations returned by the game, observation conditional probabilities, reward function, and the discount factor respectively. Drawing from this definition, each instance of our dataset takes the tuples of $\langle s_t, a_t, s_{t+1}, r_{t+1} \rangle$ where s_t and s_{t+1} are two subsequent states with a_t being the action used to transition states and r_{t+1} is the observed reward for some step t.

[2]https://github.com/microsoft/jericho
[3]https://github.com/microsoft/textworld
[4]https://parl.ai/projects/light

To collect the $\langle s_t, a_t, s_{t+1}, r_{t+1}\rangle$ tuples we implement a basic agent that explores the game along a trajectory corresponding to a *game walkthrough*. Game walkthroughs are texts describing the solutions to games, generally retrieved from the internet, but already part of the Jericho framework. Walkthroughs, however, only present one possible solution to a game and solve all the core puzzles required to complete a game with the maximum possible score. To achieve greater coverage of the game's state space, our data collection agent stops off to explore by executing random valid actions for n steps before resetting to the walkthrough. One such collected state—a part of the full tuple mentioned—is detailed below.

The textual observations consist of descriptions of the location and inventory as well as the game engine response to the previous action performed. For example:

```
Game: ztuu
Location: Cultural Complex This imposing ante-room, the center of what was apparently the cultural center
      of the GUE, is adorned in the ghastly style of the GUE's "Grotesque Period." With leering gargoyles,
      cartoonish friezes depicting long-forgotten scenes of GUE history, and primitive statuary of pointy-
      headed personages unknown (perhaps very, very distant progenitors of the Flatheads), the place would
      have been best left undiscovered. North of here, a large hallway passes under the roughly hewn
      inscription "Convention Center." To the east, under a fifty-story triumphal arch, a passageway the
      size of a large city boulevard opens into the Royal Theater. A relatively small and unobtrusive sign
      (perhaps ten feet high) stands nearby. South, a smaller and more dignified (i.e. post-Dimwit) path
      leads into what is billed as the "Hall of Science." You can see a pair of razor-like gloves here.
Observation: You put on the razor-like gloves.
Inventory:
    You are carrying:
      a brass lantern (providing light)
      a pair of glasses
      four candy bars:
        a ZM$100000
        a Multi-Implementeers
        a Forever Gores
        a Baby Rune
      a cheaply-made sword
Prev Act: put on gloves
```

We further provide the set of objects that are found in both the agent's inventory and surroundings, including textual descriptions for each of the objects. Attributes for each of these objects are also included are acquired by decompiling the games, following [Ammanabrolu et al., 2020b]. For example:

```
Inventory Objects:
    candy: Which do you mean, the ZM$100000, the Multi Implementeers, the Forever Gores or the Baby Rune?
    Implementeers: The profiles on the wrapper of this delicacy look more like Moe, Larry, and Curly than
          those of your favorite Implementeers (presumably, Marc, Mike, and David.)
    Forever/Gores: The wrapper of this bar pictures the Milky Way, but the stars are all blood red. Kids
          love them.
    sword: This is a cheaply made sword of no antiquity whatsoever. With regard to grues or other
          underworldly denizens, your weapon is as likely to engender laughter as fear.
    rune: The label is covered with mystical runes, the meanings of which elude you.
    glasses: The owner of these glasses had an indeterminate vision problem, because the lenses have both
          been crushed underfoot. The vision problem, of course, has been solved.
    lantern: The lantern, while of the cheapest construction, appears functional enough for the moment.
          Your best hope is that it stays that way. It looks like the lamp has gone through a few cycles of
           impact revitalization.
Inventory Attributes:
    glasses: clothing
    gloves: clothing
    sword: animate, equip
    lantern: animate, equip
Surrounding Objects:
    gargoyles: Unless you are inordinately masochistic, the less time spent examining the artwork, the
          better.
    east: You see nothing special about the east wall.
    tunnel: The tunnel leads west.
    gloves: The razor like gloves would be very attractive for an axe murderer. And they're just your size.
    south: You see nothing special about the south wall.
    sign: The sign indicates today's performance, which (in honor of the festivities in the Convention
          Center) is "A Massacre on 34th Street."
Surrounding Attributes:
    gloves: clothing
    tunnel: animate
    sign: animate
```

We further provide the ground truth knowledge graph representing the world state corresponding to these textual observations. The ground truth knowledge graph is a set of tuples $\langle s, r, o\rangle$ such that s is a subject, r is a relation, and o is an object. It reflects information on the current state such as objects and attributes and is extracted from the game engine by traversing the engine's internal representation and converting it to human readable form. Relations are defined on the basis of traversal operations in the game engine's internal representation, e.g. "in" and "have" signify parent-child ownership for locations and inventory respectively. For example:

```
Graph: [sign, in, Cultural Complex], [you, have, Forever Gores], [you, have, ZM$100000], [you, have, Baby
      Rune], [tunnel, in, Cultural Complex], [you, in, Cultural Complex], [you, have, brass lantern], [you,
       have, glasses], [decoration, in, Cultural Complex], [you, have, cheaply-made sword], [you, have,
      Multi-Implementeers], [you, have, razor-like gloves], [glasses, is, clothing], [gloves, is, clothing],
       [sword, is, animate], [tunnel, is, animate], [sign, is, animate], [lantern, is, animate], [sword, is,
       equip], [lantern, is, equip]
```

Valid actions are defined by Hausknecht et al. [2020] as the set of actions guaranteed to cause a change in the current world state and are identified by the Jericho framework. For example in one particular state me might have the following valid actions:

```
Valid Actions: west, turn lantern off, east, south, put multi down, put forever down, put lantern down,
      put rune down, put glasses down, put sword down, take razor off, put on glasses, examine glasses,
      lower razor, throw multi, throw lantern, put multi in glasses, north
```

3.1 Dataset Analysis

Game	No. Samples	Input Vocab Size	Avg. Obs Token Len.	Avg. Graph Triple Len.	Avg. No. Valid Actions	Avg. Surround. Objects
Training games						
wishbringer	560	1043	136.54	4.00	10.35	8.51
snacktime	168	468	190.08	2.33	4.82	5.52
tryst205	1052	871	136.24	7.81	14.30	8.38
enter	440	470	219.06	14.79	18.04	9.23
omniquest	784	460	79.96	8.02	21.50	5.30
zork3	1142	564	137.68	6.59	12.72	5.26
zork2	584	684	154.90	7.82	29.66	5.73
inhumane	1004	409	90.24	3.86	4.31	2.48
905	504	296	100.91	11.69	13.60	12.24
loose	16	1141	140.38	10.12	2.12	9.00
murdac	1914	251	80.76	4.30	8.67	1.63
moonlit	684	669	131.62	12.10	9.20	11.61
dragon	894	1049	182.79	11.64	13.13	12.29
jewel	1418	657	119.08	7.21	13.82	5.15
weapon	294	481	230.41	29.79	9.65	35.68
karn	2196	615	138.87	13.24	26.36	8.44
zenon	402	401	101.52	5.01	5.97	3.95
acorncourt	474	343	323.38	36.14	20.18	16.08
ballyhoo	2132	962	127.08	7.25	15.39	7.11
yomomma	884	619	129.06	3.00	16.11	5.52
enchanter	1714	722	133.56	14.83	45.27	7.40
gold	2082	728	166.96	15.76	25.03	12.92
huntdark	344	539	162.33	13.01	6.33	6.90
afflicted	574	762	165.13	2.91	17.34	11.85
adventureland	870	398	87.41	6.99	9.02	5.17
reverb	722	526	101.92	5.23	9.04	4.78
night	346	462	49.92	10.17	4.55	3.37
overall train	24198	11056	133.30	9.74	17.41	7.70
Testing games						
deephome	630	760	147.33	10.20	15.31	7.15
balances	990	452	107.15	7.61	13.04	3.85
ludicorp	2210	503	88.32	9.47	9.27	4.60
pentari	276	472	130.34	3.46	3.72	2.84
detective	434	344	105.97	2.80	5.72	2.16
ztuu	462	607	170.89	11.97	18.39	7.94
zork1	886	697	109.70	6.46	13.02	4.54
library	654	510	154.40	9.18	4.59	10.20
temple	1294	622	138.07	10.77	8.56	8.78
overall test	7836	11056	118.92	8.71	10.30	5.86

Table 1: Dataset statistics across the games. All games together have a combined input vocabulary of size 11056. There are 17 unique graph relations and 6985 unique graph entity names (i.e. locations, characters, and objects) across all the games. Vocabulary files are provided in the dataset.

Table 1 presents statistics for our data in the form of showing vocab sizes, and average lengths of different data fields. The dataset as a whole has an input vocabulary size of 11056—this is the superset of the vocabulary that can be used to act in any of these games. It is worth noting, that the output vocabulary size—determined by the observations—is not restricted. As seen later when we introduce models for these tasks, this means that subword based tokenization [Kudo, 2018] for processing inputs is the most effective way of avoiding unknown tokens.

The training and testing games both cover a wide range of genres as noted by Hausknecht et al. [2020], Ammanabrolu et al. [2020b]—e.g. *905* is a everyday slice-of-life simulator in which a character walks around a house preparing for work, *afflicted* is a monster horror game, *ballyhoo, detective* are murder mysteries, and *karn, zork1* are traditional fantasies. On average, the observation token, graph, and valid action lengths are comparable across both the training and testing games. Outliers in these metrics usually represent game-specific challenges. For example, *acorncourt* has the highest observation token and graph length counts by far. This is because the game is focused heavily on object collection and so contains more entities on average than others. In a similar vein, *enchanter* has significantly more valid actions than other games. This is due to the game being focused on constantly discovering valid actions in the form of spells and their effects by casting them—everything from healing yourself to causing an object to give off light. It is worth noting that many of these spells appear and have similar effects in other fantasy text games. These are some examples of challenges that players must overcome to be successful in these worlds.

4 Benchmarks

This section introduces the two primary tasks using JerichoWorld required for world modeling in learning agents: *knowledge graph prediction* and *valid action prediction*. We then introduce baseline models for each of the tasks, report zero-shot results on the testing games, and analyze performance.

4.1 Knowledge Graph Prediction

The first world modeling task involves predicting a knowledge graph from the current set of textual observations. Recall that our dataset takes the form of tuples of $\langle s_t, a_t, s_{t+1}, r_{t+1}\rangle$ where s_t and s_{t+1} are two subsequent states with a_t being the action used to transition states and r_{t+1} is the observed reward. This task is to predict s_{t+1}^{graph}, a set of knowledge graph relations, given the textual observations s_t^{obs}, the previous state's graph s_t^{graph}, and action a_t for all samples in the dataset. We present three baseline models for this task.

Rules. Following Ammanabrolu and Hausknecht [2020], we extract graph information from the observation using information extraction tools such as OpenIE [Angeli et al., 2015] in addition to some hand-authored rules to account for the irregularities of text games.

Question-Answering. This baseline comes from the Q*BERT agent described in Ammanabrolu et al. [2020b]. It is trained on the SQuAD 2.0 [Rajpurkar et al., 2018], the Jericho-QA text game question answering dataset on the same set of training games as found in JerichoWorld and then on JerichoWorld itself by formatting our dataset in the style of questions and answers when possible. It uses the ALBERT [Lan et al., 2020] variant of the BERT [Devlin et al., 2018] natural language transformer to answer questions and populate the knowledge graph via a few hand-authored rules from the answers. Examples of questions asked include: "What is my current location?", "What objects are around me?".

Seq2Seq. We further introduce a encoder-decoder based sequence-to-sequence learning approach [Sutskever et al., 2014] inspired by the transformer model BART [Lewis et al., 2020]. The model architecture is shown in Figure 3 and consists of a bidirectional encoder such as BERT [Devlin et al., 2018] that takes the full set of textual observations—including location and inventory descriptions—as input and an autoregressive decoder such as GPT-2 [Radford et al., 2019] which takes in the current graph and learns to predict the graph sequence shifted by a token. The weights of the encoder are fine-tuned from BERT's original weights on both the graphs, in triple form, and the textual observations taken from the training games using a masked language modeling loss. The decoder is not pre-trained. During test time, only the starting token is given to the decoder and it decodes the graph token by token via bean search until an end-of-sequence token is reached.

Metrics. For this task, we report two types of metrics (Exact Match or EM and F1) operating on two different levels—at a *graph* tuple level and another at a *token* level. EM checks for accuracy or direct overlap between the predictions and ground truth, while F1 is a harmonic mean of predicted precision

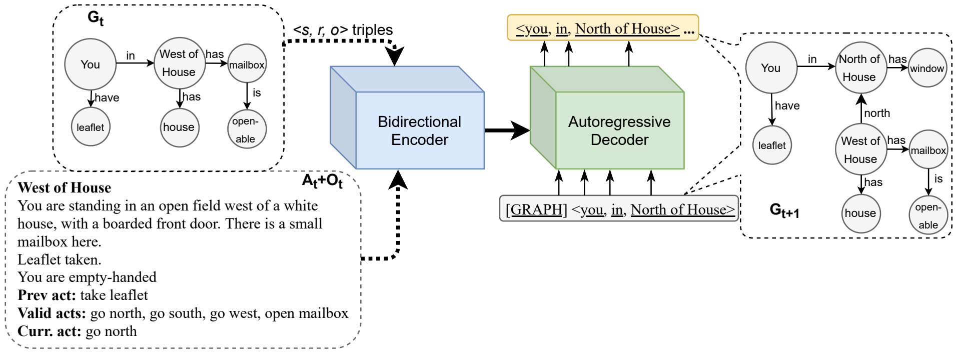

Figure 3: A description of the Seq2Seq architecture for knowledge graph prediction task with a bidirectional encoder and autoregressive decoder. A similar architecture is used for the Seq2Seq model shown in the valid action prediction task.

and recall. The graph level metrics are based on matching the set of ⟨*subject, relation, object*⟩ triples within the graph, all three tokens in a particular triple must match a triple within the ground truth graph to count as a true positive. The token level metrics operate on measuring unigram overlap in the graphs, any relations or entities in the predicted tokens that match the ground truth count towards a true positive.

Analysis. Table 2 presents a breakdown of the results for this task across the testing games on all the baseline models presented. There are a few main trends to note in these results.

The first is that the question-answering (QA) approach significantly outperforms both the Rules and Seq2Seq approaches on average across all the testing games. The QA method used is extractive. This means that the system is trained to pick out answers by highlighting spans in the input context that best answers a question. The Rules approach also functions similarly but is not trained in any way on our data. This is inherently a simpler problem formulation than the Seq2Seq approach—which seeks to generate the graph by decoding token by token—but has its limitations.

These limitations are seen in the relative differences between the magnitudes of the graph and token metrics for these approaches. Both QA and Rules have significantly lower graph metrics than token metrics, a phenomenon not observed in the Seq2Seq model. In other words, the right information is extracted but is potentially not well shaped into knowledge graph form. We hypothesize that this implies two things. (1) That these systems likely *over-extract* by extracting more information than is strictly necessary. Take for example the sample observation seen in Figure 3: *"You are standing in an open field west of a white house, with a boarded front door."*. QA when asked the question "What is my location?" answers: "open field west of a white house, with a boarded front door". Seq2Seq, in contrast, is trained to map this sentence more tersely to: ⟨*you, in, West of House*⟩. (2) Both QA and Rules use hand-crafted rules to put the graphs together once information has been extracted either through the core QA model or OpenIE. We see here that while over-extraction can be beneficial for the token metrics—it makes it difficult to create a set of graph construction rules that generalize well across games with different structures, resulting in relatively lower graph metrics.

On the other hand, the main advantage of the Seq2Seq approach is that it is not extractive and trained directly on the graphs found in the dataset. This means that it is potentially able to infer facts that are not directly present in the input context. Recall that text games are *partially observable* and so the textual observations themselves may potentially be incomplete. An example of such an observation is: *"You see a locked chest in front of you in the cellar."*. The ground truth graph for this would be: ⟨*you, in, Cellar*⟩, ⟨*chest, in, cellar*⟩, ⟨*chest, is, lockable*⟩, ⟨*sword, in, chest*⟩. The last fact in the graph, the sword being in the chest, is not revealed to you via the observation until you open the chest and thus cannot be predicted by extractive approaches like Rules and QA. This gives models like Seq2Seq—that are trained directly on the graph—the ability to perform commonsense inference by potentially filling in information missing from the partially observable text inputs. It further implies that extractive models, in their current form, will not be able to achieve perfect performance.

The main limitation of the Seq2Seq model, however, is that this non-extractive framing—given that every token is decoded autoregressively, requiring a prediction at every step over the entire combined

Expt.		Rules				Question-Answering				Seq2Seq			
Metric		Graph		Token		Graph		Token		Graph		Token	
Game	**Size**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
zork1	886	3.72	4.46	6.08	8.42	24.56	24.88	43.93	48.31	12.44	12.96	18.01	21.12
library	654	7.61	12.87	10.33	26.74	29.14	31.46	49.78	52.76	18.42	18.89	20.26	20.84
detective	434	1.39	4.55	7.51	10.23	34.45	36.23	60.28	63.21	26.86	29.48	35.86	35.86
balances	990	9.17	11.9	32.53	36.09	41.22	41.85	85.81	86.18	8.19	9.04	17.6	18.86
pentari	276	6.44	10.22	16.48	23.36	28.96	30.12	65.02	69.54	22.18	23.54	25.48	27.72
ztuu	462	4.94	10.06	14.4	21.74	22.17	26.26	49.44	49.82	16.89	16.89	17.19	17.87
ludicorp	2210	5.1	8.37	14.47	18.48	41.44	46.74	57.58	60.95	12.94	14.18	14.8	15.42
deephome	630	0.49	0.64	3.34	3.86	4.42	4.66	9.31	9.84	8.38	10.47	13.25	13.25
temple	1294	2.48	3.36	7.42	9.44	36.84	39.86	48.98	49.17	16.48	18.52	22.48	24.34
overall	7836	4.70	7.25	13.08	17.50	**32.78**	**35.48**	**53.58**	**55.74**	14.29	15.54	18.80	19.96

Table 2: Results for the Knowledge Graph Prediction task. Overall indicates a size weighted average. All experiments are evaluated over three random seeds with standard deviations not exceeding ± 2.8 in any overall category.

entity and relation vocabulary length of 7002—is a significantly more difficult problem than the other approaches. It is likely that that such non-extractive approaches will have to simplify the problem by adding constraints that account for properties of knowledge graphs (e.g. graph are sets of tuples and the same tuple in a set cannot be decoded twice).

4.2 Valid Action Prediction

The second world modeling task involves predicting the set of valid actions from the current set of textual observations. Given the data $\langle s_{t-1}, a_{t-1}, s_t, r_t \rangle$ (note the change in indexing), this task is formally defined as: predict the set of valid actions for the subsequent state s_t^{valid} given the current state text observation s_t^{obs}, current knowledge graph s_t^{graph}, previous valid actions s_{t-1}^{valid}, and action a_{t-1} that caused the state change for all individual samples across the dataset. This task requires linguistic priors in the form of commonsense reasoning and a knowledge of affordances—e.g. *open mailbox* is a more reasonable action to take in most situations than *eat mailbox*.

We present a single baseline for this task. We developed a **Seq2Seq** model that is identical to that presented for the Knowledge Graph Prediction task, except adapted to Valid Action Prediction. That is, it performs sequence learning on the valid actions token by token. Extractive approaches like QA are not possible for valid action prediction given that the verbs in the action—e.g. *take, get, put, swing, go*—are not often found anywhere within the observation. The Seq2Seq approach thus decodes actions token by token from the entire combined output vocab of 11056 (see Table 1) at every step until a special end-of-sequence tag is reached.

Metrics. For this task, we adapt the graph level Exact Match (EM) and F1 metrics as described in the previous task to actions. In other words, positive EM or F1 happens only when all tokens in a predicted valid action match one in the gold standard set. Given that most valid actions have less than four tokens, we do not use standard Seq2Seq metrics—such as BLEU [Papineni et al., 2002]—intended for measuring n-gram overlap in longer sequences. We do not report token unigram overlap, as with the knowledge graph task as here, because predicted actions are required to match gold standard actions exactly in order to be executable in the game.

Game	Size	EM	F1
zork1	886	16.65	17.85
library	654	15.13	16.88
detective	434	18.19	21.12
balances	990	16.19	18.23
pentari	276	23.39	25.87
ztuu	462	14.75	15.13
ludicorp	2210	20.1	20.86
deephome	630	14.71	14.86
temple	1294	20.34	22.14
overall	7836	18.10	19.44

Table 3: Results for the Valid Action Prediction task. Overall indicates a size weighted average. All experiments are evaluated over three random seeds with standard deviations not exceeding ± 3.7 in any overall category.

Analysis. Table 3 shows the results for the valid action prediction task on all the testing games for the Seq2Seq baseline. Recall that an EM of 20 means that if there were 100 gold standard valid actions in an instance, the model predicted 20 of them exactly. Based on this, we further note a trend in Table 3 that negative correlation between the as seen in Table 1. That is, the more the average number of gold standard valid actions per instance in a game, the more predicted actions match. Games like *ztuu, deephome, balances* have a high number of gold standard average valid actions and lower performance than games like *pentari, ludicorp, detective, temple* which have a low number of average valid actions. This is counter intuitive as the expected result would be that a model is able to learn a smaller sequence more effectively than a larger one—implying that a smaller number of gold standard valid actions per instance would lead to more matches. We hypothesize that this is likely due to the fact that the model best learns common actions found across all games first before learning potentially more fine grained actions—effectively a label imbalance issue across the valid actions in the dataset. E.g. navigation actions like *go north* are found much more often than actions like *hit monster with sword*—which are usually found in only a handful of fantasy games. When performing zero-shot prediction on testing games, the model thus predicts these common actions with higher confidence than the more fine grained ones. Testing games with a smaller number of average gold standard valid actions also tend to have a larger proportion of uncommon actions—thus posing more of a challenge for the Seq2Seq model.

5 Conclusions and Future Work

This paper presents the JerichoWorld dataset and corresponding benchmarks that seek to drive progress in textual world modeling. This primarily involves two key challenges behind the creation of agents that can understand and generate natural language in a diverse set of interactive and situated settings such as text games. Our dataset provides mappings from textual observations to ground truth knowledge graph states to enable agents to learn to infer the state of the world—alleviating the *knowledge representation* or *Textual-SLAM* challenge. A key insight from an comparison of baseline models shows that a promising future direction lies in *inferring* the knowledge graph world state through commonsense reasoning rather than *extracting* this information due to the partial observability of text games.

A second world modeling task revolves around tacking the *combinatorially-sized action space* of text games. The dataset also provides mappings from textual observations to valid actions—i.e. the set of contextually relevant actions guaranteed to change the world in any state. A qualitative analysis of a state-of-the-art Seq2Seq model adapted to the domain and trained for this task suggests that while learning to conditionally generate commonly occurring actions across a large set of games might be relatively easy, learning to generate specific and contextually relevant actions provides a significantly more difficult challenge. Current performance by state-of-the-art models across both these tasks suggests that there is much space for improvement.

There are many more tasks that can be framed for other challenges related to world modeling from this dataset. Some immediate examples: (1) offline reinforcement learning for game agents through imitation learning—predicting the sequence of actions that finish the game based on walkthroughs and reward information; (2) knowledge graph verbalization, a form of the standard data-to-text natural language processing task [Wiseman et al., 2017], in which we learn to generate text that is conditioned on a knowledge graph; and (3) description generation conditioned on the names of various objects, locations, and characters—with applications in long-form text generation domains such as automated storytelling [Martin et al., 2018, Fan et al., 2019] and procedural generation of interactive narratives [Ammanabrolu et al., 2020a, Walton et al., 2020].

6 Broader Impacts

Text games are simplified analogues for systems capable of long-term dialogue with humans, such as in assistance with planning complex tasks, and also discrete planning domains such as logistics. Our focus is on helping agents to better model such worlds, enabling greater efficiency for agents training to produce such contextually relevant language.

The data is collected from games containing situations of non-normative language usage—describing situations that fictional characters may engage in that are potentially inappropriate, and on occasion impossible, for the real world such as running a troll through with a sword. Instances of such scenarios are mitigated by careful curation of the games that the data is collected from. The original Jericho framework [Hausknecht et al., 2020]—further verified by us in this work—uses a curated set of games found not to contain extreme examples of non-normative language usage. This is based on manual vetting and (existing) crowd-sourced reviews on the popular interactive narrative forum IFDB.[5]

[5]https://ifdb.org/

References

A. Adhikari, X. Yuan, M.-A. Côté, M. Zelinka, M.-A. Rondeau, R. Laroche, P. Poupart, J. Tang, A. Trischler, and W. Hamilton. Learning dynamic belief graphs to generalize on text-based games. *Advances in Neural Information Processing Systems*, 33, 2020.

P. Ammanabrolu and M. Hausknecht. Graph Constrained Reinforcement Learning for Natural Language Action Spaces. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1x6w0EtwH.

P. Ammanabrolu and M. O. Riedl. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, 2019.

P. Ammanabrolu, W. Cheung, D. Tu, W. Broniec, and M. O. Riedl. Bringing stories alive: Generating interactive fiction worlds. In *Proceedings of the Sixteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-20)*, 2020a. URL https://www.aaai.org/ojs/index.php/AIIDE/article/view/7400.

P. Ammanabrolu, E. Tien, M. Hausknecht, and M. O. Riedl. How to avoid being eaten by a grue: Structured exploration strategies for textual worlds. *arXiv preprint arXiv:2006.07409*, 2020b.

G. Angeli, J. Premkumar, M. Jose, and C. D. Manning. Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015.

K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017. doi: 10.1109/MSP.2017.2743240.

C. Bamford and S. M. Lucas. Neural game engine: Accurate learning of generalizable forward models from pixels. In *2020 IEEE Conference on Games (CoG)*, pages 81–88. IEEE, 2020.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

M.-A. Côté, A. Kádár, X. Yuan, B. Kybartas, T. Barnes, E. Fine, J. Moore, M. Hausknecht, L. E. Asri, M. Adada, W. Tay, and A. Trischler. Textworld: A learning environment for text-based games. *CoRR*, abs/1806.11532, 2018.

S. Dambekodi, S. Frazier, P. Ammanabrolu, and M. O. Riedl. Playing text-based games with common sense. *arXiv preprint arXiv:2012.02757*, 2020.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

A. Fan, J. Urbanek, P. Ringshia, E. Dinan, E. Qian, S. Karamcheti, S. Prabhumoye, D. Kiela, T. Rocktaschel, A. Szlam, and Others. Generating Interactive Worlds with Text. *arXiv preprint arXiv:1911.09194*, 2019.

D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

M. Hausknecht, P. Ammanabrolu, M.-A. Côté, and X. Yuan. Interactive fiction games: A colossal adventure. In *Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, 2020. URL https://arxiv.org/abs/1909.05398.

D. Jancke. Orientation formed by a spot's trajectory: a two-dimensional population approach in primary visual cortex. *Journal of Neuroscience*, 20(14):RC86–RC86, 2000.

T. Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018. doi: 10.18653/v1/p18-1007.

Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1eA7AEtvS.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://www.aclweb.org/anthology/2020.acl-main.703.

L. Martin, P. Ammanabrolu, X. Wang, W. Hancock, S. Singh, B. Harrison, and M. Riedl. Event representations for automated story generation with deep neural nets. In *AAAI Conference on Artificial Intelligence*, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17046/15769.

K. Murugesan, M. Atzeni, P. Shukla, M. Sachan, P. Kapanipathi, and K. Talamadupula. Enhancing text-based reinforcement learning agents with commonsense knowledge. *arXiv preprint arXiv:2005.00811*, 2020.

K. Murugesan, M. Atzeni, P. Kapanipathi, P. Shukla, S. Kumaravel, G. Tesauro, K. Talamadupula, M. Sachan, and M. Campbell. Text-based RL Agents with Commonsense Knowledge: New Challenges, Environments and Baselines. In *Thirty Fifth AAAI Conference on Artificial Intelligence*, 2021.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language Models are Unsupervised Multitask Learners. 2019.

P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL https://www.aclweb.org/anthology/P18-2124.

C. Sautier, D. J. Agravante, and M. Tatsubori. State Prediction in TextWorld with a Predicate-Logic Pointer Network Architecture. In *In Workshop on Knowledge-based Reinforcment Learning at IJCAI-20*, 2020. URL https://kbrl.github.io/papers/08-KBRL.pdf.

J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *arXiv preprint arXiv:1911.08265*, 2019.

M. Shridhar, X. Yuan, M.-A. Cote, Y. Bisk, A. Trischler, and M. Hausknecht. {ALFW}orld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3104–3112, Cambridge, MA, USA, 2014. MIT Press.

R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998. ISBN 0262193981.

R. Tamari, F. Bai, A. Ritter, and G. Stanovsky. Process-level representation of scientific protocols with interactive annotation. *arXiv preprint arXiv:2101.10244*, 2021.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

J. Urbanek, A. Fan, S. Karamcheti, S. Jain, S. Humeau, E. Dinan, T. Rocktäschel, D. Kiela, A. Szlam, and J. Weston. Learning to speak and act in a fantasy text adventure game. *CoRR*, abs/1903.03094, 2019.

N. Walton et al. AI Dungeon. https://play.aidungeon.io/, 2020.

S. Wiseman, S. Shieber, and A. Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1239. URL https://www.aclweb.org/anthology/D17-1239.

S. Yao, R. Rao, M. Hausknecht, and K. Narasimhan. Keep CALM and explore: Language models for action generation in text-based games. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8736–8754, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.704. URL https://www.aclweb.org/anthology/2020.emnlp-main.704.