PLM-based World Models for Text-based Games

Minsoo Kim♠ YeonJoon Jung♣ Dohyeon Lee♢ Seung-won Hwang∗♢

♠ Interdisciplinary Program in Artificial Intelligence, Seoul National University
♣ Department of Computer Science and Engineering, Seoul National University
♢ Department of Artificial Intelligence, Yonsei University

(minsoo9574, waylight3, seungwonh)@snu.ac.kr
(theaitetus)@yonsei.ac.kr

Abstract

World models have improved the ability of reinforcement learning agents to operate in a sample efficient manner, by being trained to predict plausible changes in the underlying environment. As the core tasks of world models are future prediction and commonsense understanding, our claim is that pre-trained language models (PLMs) already provide a strong base upon which to build world models. Worldformer is a recently proposed world model for text-based game environments, based only partially on PLM and transformers. Our distinction is to fully leverage PLMs as actionable world models in text-based game environments, by reformulating generation as constrained decoding which decomposes actions into verb templates and objects. We show that our model improves future valid action prediction and graph change prediction. Additionally, we show that our model better reflects commonsense than standard PLM.

1 Introduction

In model-based reinforcement learning, world models (Ha and Schmidhuber, 2018), being trained to predict plausible changes in the underlying environment, has helped agents to quickly identify sensible or high-value actions. As a result, agents equipped with world models have outperformed state-of-the-art model-free algorithms in Atari games, while drastically improving sample efficiency (Kaiser et al., 2019; Hafner et al., 2020).

In this work, we focus on world modeling in text-based game (TBG) environments, in which players and agents must perceive and interact with the world entirely through textual natural language. As such, they present several unique challenges (Côté et al., 2018; Hausknecht et al., 2019; Ammanabrolu and Riedl, 2021b): Aside from language understanding itself, TBGs require dealing with an exponential action space: For instance, the action space size for Zork1, combining five words from a vocabulary of 697 words, is $O(10^{14})$. TBGs also require rich and accurate knowledge representations of locations and objects, in order to facilitate navigation and interaction. Finally, solving TBGs requires commonsense reasoning, including understanding object affordance, and the causal ramifications of actions. The first dataset and model for learning world models in TBGs were proposed by JerichoWorld (Ammanabrolu and Riedl, 2021b), and Worldformer (Ammanabrolu and Riedl, 2021a), respectively. Intuitively, large pre-trained language models (PLMs), make promising candidates for instantiating world models (Yao et al., 2020), assuming a standard vocabulary of 40,000 tokens, and context length of 512 tokens. While promising, adapting PLMs to TBGs is non-trivial, as in...
TBGs, generated actions must be executable in the game environment. While we are not generating in a formal syntax as in semantic parsing, we are still constrained by a controlled sublanguage (Shin et al., 2021), which is closer to natural language but follows a grammar defined by the engine’s parser.

Furthermore, a world modeling task such as the valid action decoding task of JerichoWorld, requires predicting, from a particular world state, all future valid actions. In contrast, semantic parsing focuses on learning a one-to-one mapping between natural language and the controlled sublanguage. To encapsulate these challenges, we define a concept of actionability as the main objective of the valid action generation task of JerichoWorld, in that the model’s generation should be actionable in the TBG environment, by 1) conforming to the parseable controlled sublanguage, and 2) ensuring the generation of valid actions consistent with the commonsense governing the dynamics of the TBG environment. In this work, we focus on improving the actionability of the world model’s generations, focusing in particular on bringing the PLM’s commonsense and reasoning capabilities to bear on world modeling tasks.

We begin by building Worldformer-BART, an implementation of Worldformer based on BART. While successful, commonsense in PLM-based world models are not easily transferred to more formal forms, such as that of the controlled sublanguage of TBGs. Motivated by these findings, we build a retrieval-augmented model which aims to enhance actionability by formalizing commonsense through templates, hence named Actionable World Model (AWM). We show through experimental results that AWM-BART significantly improves actionability over Worldformer-BART. We also compare with augmenting trained models with a COMET-based commonsense filter, showing that our model outperforms such approaches.

2 Background

2.1 Text World Environments

Text world environments are typically modeled as Partially-Observable Markov Decision Processes (POMDPs) (Côté et al., 2018; Hausknecht et al., 2019), defined by the tuple \((S, A, T, \Omega, O, R, \gamma)\). Respectively, each item in the tuple corresponds to the set of environment states, the text-based action that changes the game state according to a transition function, the mostly-deterministic latent transition function, observation conditional probabilities, the observations, i.e. the game’s text responses, the reward function, and the discount factor.

While state-of-the-art agents can be trained on these environments using model-free RL algorithms, they often rely on large amounts of interaction with the environment (Kaiser et al., 2019; Yarats et al., 2021). In contrast, model-based learning proposes to learn a predictive model of the environment, also known as a world model, to aid an agent to learn the underlying dynamics of the game, and better predict which actions will lead to desirable outcomes. These approaches are closely inspired by research on human cognitive processes, which hypothesize that human decision-making is directly influenced by an internal predictive model of the future (Ha and Schmidhuber, 2018). For world models, recurrent neural networks are a suitable solution to overcome the partial observability of the environment in POMDPs, and in text-based games environments, pre-trained language models are promising candidates.

Popular text-based game environments include TextWorld (Côté et al., 2018), which provides procedurally generated environments, allowing for the complexity and content of the generated game to be variable, LIGHT (Urbanek et al., 2019), a large-scale crowdsourced text adventure game, whose dataset provides agent-to-agent dialogs to study grounded social interactions, and Jericho (Hausknecht et al., 2019), a collection of 32 diverse human-made interactive fiction games, covering a wide range of genres.

2.2 JerichoWorld

Different from the aforementioned environments, JerichoWorld (Ammanabrolu and Riedl, 2021b) is the first dataset specifically targeting the learning of world models in text-based game environments. JerichoWorld is generated by simulating playthroughs of Jericho games, based on human-generated gold walkthroughs, combined with random exploration to increase the coverage of the state spaces of games. Each example in the dataset is a tuple of the form, \((S_t, A_t, S_{t+1}, R)\) consisting of a previous state \(S_t\), a transition action \(A_t\), the next state \(S_{t+1}\), and the observed reward \(R\). A key feature of the dataset is that it maps text observations to both knowledge graphs, which consist of a set of \((s, r, o)\) tuples that reflecting the world state, and a set of valid actions. Valid actions in Jericho...
Table 1: Illustration of the inputs and generated outputs of the valid action prediction task, on Zork3. The results are from a Worldformer-BART model.

<table>
<thead>
<tr>
<th>Observation</th>
<th>Graph</th>
<th>Valid actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dropped. Location: Great Door. This is the south end of a monumental hall, full of dust and debris from a recent earthquake. To the east is a great iron door, rusted shut. To its right, however, is a gaping cleft in the rock and behind, a cleared area. There is a torch here. You are carrying: • A cloak (being worn) • A hood (being worn) • A vial • A wooden staff • A strange key • A Frobozz Magic Grue Repellent • A golden amulet (being worn)</td>
<td>you at Great Door, 'you have key', 'you have hood', 'torch in Great Door', 'you have golden amulet', 'you have Frobozz Magic Grue Repellent', 'you have cloak', 'heavy invisible liquid in vial', 'you have staff'</td>
<td>• blow out torch • take torch • put down magic • put down key • put down staff • put down amulet • put down vial • put down hood • put down cloak • put down all • open vial • light magic with torch • light staff with torch • east • north</td>
</tr>
</tbody>
</table>

The textual observation consists of the engine response to the previous action, and the engine response to the commands, “look”, and “inventory”.

the transition action $A_t$. For the graph prediction task, we follow the simplification in Ammanabrolu and Riedl (2021a), to limit the task to predicting node additions, i.e. predicting the graph difference caused by additions, as these are sufficient to infer node deletions as well.

The second task is to predict the set of future valid actions at time step $t + 1$, $V_{t+1} \in S_{t+1}$, from $O_t, V_t, G_t \in S_t$ and the transition action $A_t$. In this work, we focus on the valid action prediction task, which we illustrate in detail in Table 1.

2.2.2 JerichoWorld 2.0

Since JerichoWorld is generated by simulating Jericho games, to ensure there are no data artifact issues, we also provide an updated version of JerichoWorld. We follow the methodology in Ammanabrolu and Riedl (2021b) to generate the data, making sure to generate roughly the same number of instances for each game, from the same human walkthroughs. Overall, there are 24,198 training instances and 7505 test instances in our dataset. A comparison of the test sets can be found in Appendix A.4. Our dataset additionally provides templates and objects for all actions, as well as de-abbreviating commands from the human playthrough for consistency of commands.

3 Preliminaries

We now describe our base models in detail. We begin with a description of Worldformer, which forms the basis of our model architectures.

3.1 Background: Worldformer

Worldformer is a multi-task architecture designed to perform the dual world modeling tasks of JerichoWorld. The model consists of two BERT-based encoders and two randomly initialized transformer decoders. The first pair of encoder and decoder solves the future valid action prediction task, and the second solves the future graph prediction task. Both tasks are learned simultaneously via multi-task learning. Additionally, a domain-adaptive MLM task is used to further pre-train the encoders, and a domain-specific vocabulary and tokenizer built for Jericho is used for both encoders and decoders. Ammanabrolu and Riedl (2021a) show that Worldformer achieves state-of-the-art performance on both tasks.
3.2 Baseline: Worldformer-BART

As a starting point for adapting PLMs as world models, we build a world model based on adapting BART with minimal changes. The model consists of two pre-trained BART models, arranged in a similar multi-task architecture to Worldformer. We fine-tune both BARTs directly, without any further pre-training or any changes to vocabulary or tokenization, such as building a domain-specific vocabulary as in Ammanabrolu and Riedl (2021a), or limiting the softmax operation. We give a full description of the model below. For ease of notation, we divide BART into its encoder and decoder components, where BART_{enc} and BART_{dec} combine compose the first BART encoder-decoder, and BART_{graph}_{enc} and BART_{graph}_{dec} compose the second.

Formally, the JerichoWorld example \( \langle S_t, A_t, S_{t+1}, R \rangle \) yields two sets of input and target token sequences, \( X = \{x_1, ..., x_n\} \) and \( Y = \{y_1, ..., y_n\} \). That is, for the valid action prediction task, \( \langle X, Y \rangle = \langle O_t + V_i + A_t, V_{t+1} \rangle \), and for the graph prediction task, \( \langle X, Y \rangle = \langle G_t + A_t, G_{t+1} \rangle \) where \( + \) indicates string concatenation. During model operation, each BART encoder produces contextual encodings given their respective inputs, \( O_t = \text{BART}_{\text{enc}}(O_t + V_i + A_t) \) and \( G_t = \text{BART}_{\text{graph}}(G_t + A_t) \). As in Worldformer, the aggregation module produces the state vector, \( s_t = \text{Agg}(O_t, G_t) \).

Without loss of generality, both tasks are modeled by the conditional distribution,

\[
P(Y|X) = \prod_{i=1}^{n} P(y_i|y_{<i-1}, s_t; \text{BART}_{\text{enc}}(X))
\]

and each BART encoder-decoder is trained to generate the target sequence through maximum log likelihood loss, as follows:

\[
L_{\text{gen}} = \log P(Y|X) = \sum_{i=1}^{n} \log P(y_i|y_{<i-1}, s_t; \text{BART}_{\text{enc}}(X))
\]

where \( p \) is modeled by corresponding \( \text{BART}_{\text{dec}} \).

3.3 Motivation: Qualitative Study

To motivate actionability objectives, we qualitatively analyze Worldformer-BART, which is an effective starting point for building actionable world models, but with three major types of actionability errors:

- **Object Localization and Inference (OLI) errors**: We define these as reasoning errors wherein the model fails to track the current location of object(s), i.e., whether an object is found in the inventory, the surrounding environment, or is not found at all. Examples include attempting to put a first aid kit down, before ever having picked it up, or trying to open a case of cigarettes after it has been removed from the player’s possession.

- **Object Affordance errors**: These occur when an incorrect understanding of the affordance of objects leads the model to generate actions which are nonsensical or impossible. Examples include asking a library about a library, attempting to drink out of an empty bucket, or looking with a net, etc. These also constitute false positive generated actions.

- **Insufficient Interaction Coverage**: We define these errors as those in which, despite the presence of objects, the model’s generation is insufficient to enumerate all possible interactions with the objects. These errors can be caused by errors of reasoning, or by the inability of decoding schemes to generate with high coverage. Most false negatives fall into this category.

We perform human analysis of 50 randomly sampled examples from the validation set. A subset of samples from the human analysis can be found in Appendix A.7. Of the analyzed samples, we find that ~33%, ~48%, and ~88% of samples exhibit each type of error, respectively. These results indicate that actionability for world models is not sufficiently satisfied by naive adaptation of PLMs.

4 AWM-BART for Actionable World Model

We now propose our model, which aims to improve the actionability of the BART-based world model. We decompose action generation as template selection and filling, aiming to capture two benefits: Through input-constrained decoding using templates, we enhance the parseability of the world model’s generations. Furthermore, we posit that the inductive bias from templates will aid the world model to learn more accurate object affordance and object localization, improving commonsense.
More specifically, $\text{BART}_{\text{action}}$ now performs the task of generating the sequence of possible fillings $Y_{tj} = \{y_{tj}^1, ..., y_{tj}^m\}$ of a template $t_j \in T_{\text{env}}$, of a Jericho game environment. Then, the template-conditional action generation task is reformulated as follows:

$$
P(Y_{tj} \mid X, t_j) = \prod_{i=1}^{n} P(y_{tj}^i \mid y_{tj}^{i-1}, s_t; \text{BART}_{\text{enc}}(O_t + V_t + A_t + t_j))$$

The mask filling task is illustrated in Fig. 1. The above formulation is advantageous in that the task of filling the masks of an action template brings the generation objective close to BART’s original objective. However, it does not address the issue of choosing an appropriate template to fill. We next describe how we employ multitask learning to utilize a single BART encoder-decoder as both a template retrieval and generation model.

### 4.1 Template Retrieval

In contrast to previous works utilizing template selection and filling (Hausknecht et al., 2019; Ammanabrolu and Hausknecht, 2020) for in-domain learning of agents, our aim is to build a model to generalize to any arbitrary set of natural language templates, as the world modeling tasks of JerichoWorld require zero-shot prediction on unseen games. Taking inspiration from recent advances in retrieval using neural models, we propose to extend $\text{BART}_{\text{enc}}$ as a retriever. The goal of the retriever is to identify the subset of templates $T_{t+1}^{\text{valid}} \subseteq T_{\text{env}}$, i.e. the templates defining the valid actions $V_{t+1}$, given the current world state $S_t$ and the transition action $A_t$. Therefore, we use the set of future valid actions, $V_{t+1}$, to extract the valid templates $T_{t+1}^{\text{valid}}$.

In the retrieval nomenclature, first- and second-stage retrieval refer respectively to a fast and efficient retrieval model to quickly identify a set of promising candidates, and a computationally expensive but effective re-ranking model, which produces fine-grained rankings over the smaller set of first-stage candidates. In our case, it is possible to adopt $\text{BART}_{\text{enc}}$ as a dense first-stage retriever (Lee et al., 2019; Karpukhin et al., 2020), as well as a re-ranker (Nogueira et al., 2019). Note that in our setting, since we have access to the set of all possible templates, and their number does not exceed 300 for any environment, we forego the usage of a first-stage retriever, and simply enumerate over all templates (on average around ~200).

To learn the relevance score $r_j$ between $\langle O_t, V_t, A_t \rangle$ and each template $t_j \in T_{\text{env}}$, for each template $t_j$ we concatenate $t_j$ to the encoder input. The BART encoder operates as before, but we now extract a summary vector as well as the contextual encodings, i.e. $(O_t, o_t) = \text{BART}_{\text{enc}}(O_t + V_t + A_t + t_j)$. Here, $o_t$ can be any pooled vector, and we choose the vector encoding of the EOS token. Similarly, we extract $g_t$ as $(G_t, g_t) = \text{BART}_{\text{graph}}(O_t + V_t + A_t)$. These vectors are concatenated with state vector $s_t$, and fed to a re-ranking head, which computes the template relevance score $r_j$:

$$
r_j = P(l_j = 1; o_t \oplus g_t \oplus s_t)$$

where $l_j$ indicates the ground truth label of template $t_j$. The re-ranking cross-entropy loss is defined as follows:

$$
\mathcal{L}_{\text{rerank}} = - \sum_{i \in I_{\text{pos}}} \log(r_i) - \sum_{i \in I_{\text{neg}}} \log(1 - r_i)
$$

where $I_{\text{pos}}$ are the indices of templates in $T_{t+1}^{\text{valid}}$, and $I_{\text{neg}}$ are the indices of templates belonging to the complement set. The re-ranking objective enables the full utilization of the BART encoder to model fine-grained relationships between the input context and each template.

### 4.2 Template Filling

In addition to the template retrieval task, given template $t_j$, we use $\text{BART}_{\text{action}}$ to generate the filled version of the template, with $P$ defined in Eq. 3.

$$
\mathcal{L}_{\text{fill}} = \log P(Y_{tj} | X, t_j)
$$

As per the definition in Eq 3, the same $\text{BART}_{\text{enc}}$ is shared between both retrieval and filling tasks, allowing efficient adaptation of BART to the target controlled sublanguage through multitask learning. We observe that conditioning the encoder alone with templates can effectively force the decoder to produce faithful fillings of the provided masked template. While template constraints intuitively improve the actionability of model generations in terms of parseability, we additionally expect that templates can provide a useful inductive bias for reducing OLI and Object Affordance errors. Our hypothesis is that, since the decoder is trained to fill only a single masked template at a time, this
has the implicit effect of marginalizing out the effect of other templates. To see why, consider the default decoding objective which treats all valid actions (generated from all \( t_j \in T_{t+1}^{\text{valid}} \)) as a single sequence to be generated. This causes every action to be conditioned on other valid templates \( t_j \in T_{t+1}^{\text{valid}} \) which appear at a previous position in the target sequence. In contrast, our template conditioned objective removes this effect, replacing it with a strong conditioning on the masked template, allowing the affordance relationship between template verbs and their corresponding objects to be learned efficiently. Finally, during inference time, the template retrieval module works as an effective filter which refines the action space to a promising subset, further reducing the room for error.

### 4.3 Training

In our experiments, we found it most effective to train the model in phases. In the first phase, we multi-task train the template filling task for action generation, together with the graph prediction task:

\[
\mathcal{L}_{\text{phase1}} = \mathcal{L}_{\text{fill}} + \mathcal{L}_{\text{gen}}
\]

Then, using the trained weights, we train on all losses simultaneously in the second stage:

\[
\mathcal{L}_{\text{phase2}} = \mathcal{L}_{\text{fill}} + \mathcal{L}_{\text{gen}} + \mathcal{L}_{\text{rerank}}
\]

Note that, when training in the second phase, batch items vary depending on whether the template \( t_j \) in the input \( O_t, V_t, A_t, l_t \) has label \( l_j = 1 \), or \( l_j = 0 \). We activate the full loss only in the former case, and only activate \( \mathcal{L}_{\text{rerank}} \) in the latter via a loss mask.

### 4.4 Hard Negatives Mining

Our retrieval formulation motivates our application of the technique of hard negative example mining to the template retrieval task. We supply hard negatives from a model trained with Eq. 8, as additional negative examples for Eq. 5. As we later show, hard negatives further improve the accuracy of reranking, and the overall performance of valid action generation.

### 5 Experiments

We evaluate our models on the JerichoWorld modeling tasks.

#### 5.1 Metrics

We report the F1 and EM metrics from Ammanabrolu and Riedl (2021b), where F1 is a harmonic mean of predicted precision and recall, while EM (exact match) checks for accuracy or direct overlap between the predictions and ground truth. The original dataset defines F1 and EM at two different levels: token-level, and graph tuple-level.

We focus on the tuple-level metrics, as they are the main metrics for the action task in Ammanabrolu and Riedl (2021a). For the graph task, a tuple-level true positive occurs when all three items within an \( \langle s, r, o \rangle \) tuple\(^3\) matches a tuple within the ground truth graph. The same holds for the action task, where predicted and ground truth valid actions are likewise defined as ordered tuples of tokens. The tuple-level metrics are stricter and more relevant for actionability, as EM match means the model generated an action correctly in its entirety, making the action executable by the game engine.

#### 5.2 Baselines

We compare AWM-BART with the following models:

1. **Worldformer-BART Action decoder from scratch** is our reimplementation of Worldformer (Ammanabrolu and Riedl, 2021a). To reproduce Worldformer, which is a multi-task world model composed of pre-trained BERT encoders and transformer decoders, We initialized the action decoder weights from scratch, while keeping the rest of the BART pre-trained weights. Under such minor modification, we achieved similar results reported in original Worldformer.\(^4\)

2. **CALM** (Yao et al., 2020) is a GPT-2 based model finetuned to generate \( V_{t+1} \) from \( O_t, A_t, O_{t+1} \). While it does not directly train on the Jericho suite of games, it trains on a dataset of 426 human gameplay transcripts for 590 different text-based games, ClubFloyd\(^5\).

3. **Worldformer-BART** is our implementation of Worldformer based on BART, a simple adaptation of PLMs as world models.

4. **Worldformer-BART + COMET\(^6\)** is a COMET-augmented version of Worldformer-BART. We use logits of a pre-trained COMET model to filter out

\[^{3}\text{We do not use separator tokens within each } \langle s, r, o \rangle \text{ tuple in the graph generation task.}\]

\[^{4}\text{Original Worldformer results can be found in Appendix A.3}\]

\[^{5}\text{http://www.allthingsjacq.com/interactive_fiction.html}\]

\[^{6}\text{Implementation details in Appendix A.5}\]
actions, based on the commonsenseness of actions given the current observation and inventory.

6 Results
We report the results of our experiments in Table 2. Our Worldformer-BART performs on par with Worldformer on graph prediction, but shows improvement on action generation, due to the leveraging of BART. Augmenting Worldformer-BART with a COMET-based filter fails to improve the action generation performance meaningfully, indicating that zero-shot adaptation of conventional commonsense PLMs to TBGs is challenging. Compared to CALM, our model is significantly better in action generation, indicating the importance of considering actionability in adapting PLMs to TBGs. Our model was able to outperform all compared models in both tasks, achieving a significant improvement in action generation.

6.1 Ablation Study
In order to validate the usefulness of each of our model components, we report the results of the ablation study in Table 3. We begin by comparing Worldformer-BART and AWM-BART trained with Eq. 8 without hard negatives (AWM-BART - hard negatives), where we observe that our proposed template-constrained architecture improves the learning of both tasks over Worldformer-BART. We next compare the two AWM-BART variants with and without hard negatives, which shows that while maintaining graph prediction performance, hard negatives significantly boost valid action generation, by making the reranking head more robust and reducing false positives. Finally, we report the results from using an oracle template retriever with AWM-BART (AWM-BART + Oracle). We can see that our trained template retriever approaches the performance of the oracle retriever, which indicates that there are potentially more improvements to be gained by improving the decoder.

6.2 Commonsense Study
In order to scale the analysis from Sec. 3.3 to the entire test set, we build an automated, rule-based system for detecting the OLI and affordance errors. We validate the system on the human-annotated samples, where the system recovered 91% of the human-annotated OLI and affordance errors. Implementation details are provided in Appendix A.6. In Table 4, we report the results of the commonsense error analysis using the automated system. The purpose of this study is to determine whether the improved performance of AWM-BART is attributable to the model's improved commonsense understanding. We compare our model with Worldformer-BART, and Worldformer-BART with a COMET filter.

We find that the number of errors successfully filtered out by COMET was negligible, and was outpaced by the increase in the number of false negatives. In contrast, relative to Worldformer-BART, AWM-BART reduces OLI errors by ~47%, affordance errors by ~33%, and insufficient generation coverage errors by ~24%. Taken in conjunction with the results on world modeling tasks,

Table 2: Results on JerichoWorld world modeling tasks. We report experimental results on JerichoWorld 2.0. Best overall results are denoted in bold. Asterisk (*) denotes statistically significant (P<.001) improvement over Worldformer-BART using a paired t-test. All results are averaged over three random seeds, with standard deviation under ±1.60 in the overall categories for either task.
Table 3: Results of ablation experiments. All results are averaged over three random seeds, with standard deviation under ±1.60 in the overall categories for either task.

Table 4: Number of errors per type for compared models as measured by automated system. Parentheses indicate the percentage reduction of each type of error, with respect to Worldformer-BART. All results are averaged over three random seeds.

these findings lend support to our hypothesis that the template-constrained generation of our model enhances not only the parseability in a target controlled sublanguage, but improves the commonsense understanding of the PLM as a world model. While the results of AWM-BART are encouraging, our findings raise new questions about precisely what kind of commonsense is being captured by PLMs, and whether the conventional definition of commonsense in the literature is general enough to capture the varied ways in which humans can employ commonsense, as they do in TBGs.

7 Related Works

7.1 Template-based Action Space

For in-domain generalization to a single-game setting, previous works such as LSTM-DQN (Narasimhan et al., 2015), TDQN (Hausknecht et al., 2019), and KG-A2C (Ammanabrolu and Hausknecht, 2020) have proposed using template-based action spaces in text-based games. Our distinction of employing templates, is to improve actionability of PLM-based world models. In particular, we design a template retrieval task which allows our model to generalize to any arbitrary set of templates, unlike previous works. Template retrieval plays a key role in ours, for generalizing to unseen, out-of-distribution games.

7.2 PLMs for TBG

While research adapting PLMs for TBGs is still in its early stages, there are notable works which have leveraged the linguistic, semantic and commonsense priors of PLMs to TBG agents. DBERT-DRRN (Singh et al., 2022) proposes a single-game agent based on DistilBERT (Sanh et al., 2019) to improve the semantic understanding of agents, achieving state of the art on performance on several games in Jericho. CALM (Yao et al., 2020) trains a GPT-2 model to train a single model which can be deployed to generate actions across many different downstream games, and CALM is shown to improve the existing agents on unseen games by reranking action candidates to maximize rewards. Like CALM, we tackle generalizing PLMs to unseen games, but our distinction is to consider actionability as the key criterion in doing so. Our results show that actionability is indeed crucial in building PLM-based world models for TBGs.

7.3 Commonsense in Text-based Games

Prior works have proposed incorporating external commonsense knowledge to TBG agents. Dambekodi et al. (2020); Ryu et al. (2022) propose to incorporate commonsense knowledge into a KG-A2C agent by augmenting its graph with commonsense inferences using COMET (Bosselut et al., 2019). Murugesan et al. (2021) propose to
jointly leverage a commonsense knowledge graph directly retrieved from ConceptNet along with a textual graph. Ammanabrolu and Riedl (2019) propose to use knowledge graphs to transfer commonsense across agents. Our distinction is to envision and enhance commonsense as an innate component of TBG world models.

8 Conclusion

In this work, we build world models based on PLMs. Towards leveraging semantic and linguistic priors learned by PLMs, we identified major areas in which PLMs need to be systematically improved, namely generation to a controlled sublanguage, and commonsense understanding. We propose an actionable world model based on template-augmented generation, showing that both parseability and commonsense understanding can be significantly improved. As future work, we consider combining PLM-based world models with reward-based agents to learn goal-directed policies as a promising direction. Finally, our hope is that our work will contribute to the active exploration of text-based games as an alternative testbed for further research on commonsense reasoning.

Limitations

A limitation of our work is that while we aim to adapt pre-trained language models as world models for text-based game environments, due to limited computational resources, we are not able to test our models on larger scales, and take greater advantage of language model scaling laws (Kaplan et al., 2020). As such, our results should be understood in terms of the intermediate-scale regime of PLMs. Nevertheless, we believe that actionable world models should keep efficiency as one of their core criteria.

A second limitation is that in this work, our experiments have focused solely on data which originates from the Jericho suite of games. While this is outside the immediate scope of this work, there are several other TBG environments which may be suitable candidates for learning PLM-based world models. While scaling to a greater number of TBG environments is a challenging task, we hold the view that the promise of world models in TBGs will be fulfilled by models which generalize to many environments across varying domains, structures and rules.

Acknowledgements

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) [NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)] and [(No. 2022-0-00077, AI Technology Development for Commonsense Extraction, Reasoning, and Inference from Heterogeneous Data)].

References


Antoine Bosselut, Ronan Le Bras, , and Yejin Choi. 2021. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question


A Appendix

A.1 Model Architecture details

We use BART-base (Lewis et al., 2020) to build Worldformer-BART and AWM-BART. The overall parameters counts of the models are 320 million and 322 million, respectively, indicating a roughly ~15% reduction relative to Worldformer, which has 380 million parameters. We use BART’s default tokenizer with the exception of adding several special tokens for usage as delimiters. All models generate with beam search decoding, using a beam size of 15.

A.2 Training details

We use Adam optimizer (Kingma and Ba, 2015), with learning rate $3 \times 10^{-5}$ and batch size 16 to train our models. For AWM-BART, we first train with Eq. 7 for a single epoch, then continue training with Eq. 8, reducing learning rate by 1/10. For the latter phase, to train the retrieval head, we randomly sample negative templates with 1:1 ratio between positive and negative templates. When applying additional hard negative mining, the mined templates are added to the pool of randomly sampled negatives, increasing the total number of negative templates. We perform model selection based on the combined loss of all tasks on the validation set. Our models are trained using a single NVIDIA GeForce RTX 3090 Ti GPU, taking between 3 and 5 hours per epoch.

A.3 Original Worldformer Results

As a reference, report the results reported in the original Worldformer, on the original JerichoWorld dataset in Table 5.

A.4 JerichoWorld 2.0 Test set details

In Table 6, we compare the the test sets of JerichoWorld 2.0 with that of the original, to confirm that the distribution of instances across of games remains consistent.

A.5 Implementation COMET-based Score Filter

We adopt an off-the-shelf COMET-BART model (Hwang et al., 2021) as a commonsenseness scorer. We follow the method in Bosselut et al. (2021); Ryu et al. (2022) to compute the score of a target sequence of tokens, based on COMET’s logits. We feed the current observation context and inventory, concatenated with the transition action and an ATOMIC2020 relation as the input to COMET, as done in Ryu et al. (2022). Each token in the target sequence, which is an action being evaluated, is then fed sequentially to COMET, to obtain model logits. We use the relations xNeed and xWant, averaging the two scores for each token, and the overall score of the action is given as the average over the tokens in the action. To filter out actions, we compute the mean and standard deviation of scores of all generated actions per data example, and drop outliers whose scores are more than 1 standard deviation lower than the mean.

A.6 Implementation of Automated Commonsense Error Detector

Based on our preliminary analysis, we develop a heuristic method for detecting commonsense errors defined in Sec. 3.3. We leverage the $(S_t, A_t, S_{t+1})$ tuples in the dataset, to directly measure these er-
errors automatically. Specifically, we detect the following types of errors:

- **Object Localization and Inference errors**: For false positive actions, 1. Predicted valid action attempts to interact with an object not present in the current environment or inventory. 2. Predicted valid action attempts to take object from the environment when it is expected to be in inventory already, after the transition action. 3. Predicted valid action attempts to put object down when object is not expected to be in inventory after the transition action. Furthermore, we build special heuristics for when the object in question is ‘all’, as the game’s parser understands this as referring to all objects in either the environment or the player inventory, depending on the action.

- **Object Affordance Errors**: After first filtering for OLI errors, we check an action for the following: First, we check that a false positive action has at least two noun phrases, such that an affordance between two objects can be established. Next, to account for idiosyncrasies of the engine in terms of action validity (e.g. *push lens to glasses* is a valid action), we check that the action was not a valid action in the previous state. If the action passes both checks, we judge the action to be an affordance error (e.g. *throw lantern at garlic*)

- **Insufficient Interaction Coverage**: We consider all false negatives as these.

To extract objects in predicted actions, we extract noun phrases using the Flair framework (Akbik et al., 2018). To check the existence of objects in the environment, inventory and graph, we use simple string matching of the object name.

### A.7 Qualitative study samples

In Tables 7 through 9, we show a subset of the dev set error analysis results, comparing the human annotated errors with the annotation by our automated system.

### A.8 Examples of model predictions

In Tables 10 through 14, we show several examples of models’ valid action predictions on the test set. Worldformer-BART and AWM-BART are compared.
This area of matted-down crabgrass lies between the vaulted big top entrance to the north and the enticements of the midway to the east, where a sagging banner hangs crookedly above a turnstile. There is a drinking fountain near the side wall of the tent. You can enter the night to the west and south. [OBS] You emerge into the warm night air of summer.

Connection [TRANSITION ACT] search balloon

You have a clown mask, a fiberglass pole, a balloon and $12.81 to your name.

The passage bends from northwest to east and there is a flight of steps down at this point. There is a string of shiny glass beads here. There is a dull toadstone here.

[OBS] You are holding: A lamp (which is on)

[TRANSITION ACT] take all

[OBJECT] close balloon

[ACTION] hit balloon

[ACTION] drop all

[ACTION] drop balloon

[ACTION] drop pole

[ACTION] drop mask

[ACTION] drop lamp

[ACTION] east

[ACTION] north-west

[ACTION] down

Table 7: Errors analysis of a validation set example from Ballyhoo: OLI errors occur because there is no passage or bucket in the environment.

Table 8: Errors analysis of a validation set example from Loose. The action 'take all' will not cause a change in the environment after taking transition action 'take all'.
Observation: Blue Room In the far corner of this tented enclosure a thick, undulating cloud of smoke hovers over a poker game. Straight across from you a tight-jawed dealer stands over a blackjack table covered with a green floor-length tablecloth. The spring-loaded secret panel slides shut.


Table 9: Errors analysis of a validation set example from Loose.
Table 10: Comparison of model predictions on a test set example from Ludicorp.

<table>
<thead>
<tr>
<th>Observation</th>
<th>Graph</th>
<th>Inventory</th>
<th>Valid actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>[OBS] Location: Copier Room</td>
<td>The copier room doesn’t contain any windows, and vibrates slightly with fluorescent light. A big copier sits quietly in the corner. Doors lead east and west. You can see a Dragon Statue here.</td>
<td>[TRANSITION ACT] put wire down</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[OBS] Copier Room You can see a Dragon Statue here.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[TRANSITION ACT] put wire down</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 11: Comparison of model predictions on a test set example from Balances.

<table>
<thead>
<tr>
<th>Observation</th>
<th>Graph</th>
<th>Inventory</th>
<th>Valid actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>[OBS] Location: Cave Mouth</td>
<td>This is a cave mouth, at one end of a road which winds southeast over rising ground. The entrance west to the caves is a dark tunnel, and only a footpath runs further north, into gorse. The iron door stands open. You can also see a silver coin and a spell book here.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[TRANSITION ACT] close door</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[TRANSITION ACT] close door</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>take all , close paper tray , 'west' , 'east'</td>
<td>take key</td>
<td>take wire , 'take statue' , put ladder down' , 'put gun down' , 'put all down' , 'empty paper tray' , 'take all off paper tray'</td>
</tr>
</tbody>
</table>

Worldformer-BART Predictions

<table>
<thead>
<tr>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>east , 'take all', 'take statue' , 'take wire' , 'put key down' , 'put all down' , 'empty paper tray' , 'close paper tray' , 'west' , 'take all off paper tray'</td>
<td></td>
<td>put ladder down' , 'put gun down'</td>
</tr>
</tbody>
</table>

AWM-BART Predictions

<table>
<thead>
<tr>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>east , take all , 'take statue' , 'take wire' , 'put key down' , 'put all down' , 'empty paper tray' , 'close paper tray' , 'west' , 'take all off paper tray'</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inputs to the Model
Observation
[OBS] Location: Torch Room
This is a large room with a prominent doorway leading to a down staircase. Above you is a large dome. Up around the edge of the dome (20 feet up) is a wooden railing. In the center of the room sits a white marble pedestal. A piece of rope descends from the railing above, ending some five feet above your head. Sitting on the pedestal is a flaming torch, made of ivory. [OBS] Torch Room
This is a large room with a prominent doorway leading to a down staircase. Above you is a large dome. Up around the edge of the dome (20 feet up) is a wooden railing. In the center of the room sits a white marble pedestal. A piece of rope descends from the railing above, ending some five feet above your head. Sitting on the pedestal is a flaming torch, made of ivory. [TRANSITION ACT] put down knife

Valid actions

[TRIPLE] you have brass lantern
[TRIPLE] pedestal in Torch
[TRIPLE] you have clove garlic
[TRIPLE] you have nasty knife
[TRIPLE] you have sword
[TRIPLE] torch in pedestal
[TRIPLE] you in Torch
[TRANSITION ACT] put down knife

[OBS] Inventory: You are carrying:
A sword
A nasty knife
A brass lantern (providing light)
A clove of garlic


[OBS] Inventory: You are carrying:
A sword
A nasty knife
A brass lantern (providing light)
A clove of garlic

[WORLDFORMER-BART Predictions]
True Positives
put out lantern, 'take torch', 'south'

False Positives
'take sword', 'put sword to marble', 'throw lantern at sword'

take knife, 'take all', 'put down lantern', 'put down garlic', 'put down sword', 'put down all', 'put garlic on marble', 'throw lantern at knife'

[AVM-BART Predictions]
True Positives
put out lantern, 'throw lantern at knife', 'take knife', 'south', 'take torch', 'put garlic on marble', 'put down all', 'put down lantern', 'take all'

False Positives
put knife on marble

put down garlic, 'put down sword'

Table 12: Comparison of model predictions on a test set example from Zork1.
Inputs to the Model

Observation | Graph | Inventory | Valid actions
--- | --- | --- | ---
[OBS] Location: West Royal Road
This road is quite beautiful, decorated on its sides with fluorescent mosses that feed on the minerals in the stones that line the sides of the roads. Somehow, the mosses do not leave their designated stones. High walls on both sides make the street feel more like a hall than an open passage-way, and gates leading to palaces break up the monotony of the stone. A single gate is open to the north. The road continues east and to the west is the outer court of the Lord’s Palace. [OBS] West Royal Road
This road is quite beautiful, decorated on its sides with fluorescent mosses that feed on the minerals in the stones that line the sides of the roads. Somehow, the mosses do not leave their designated stones. High walls on both sides make the street feel more like a hall than an open passage-way, and gates leading to palaces break up the monotony of the stone. A single gate is open to the north. The road continues east and to the west is the outer court of the Lord’s Palace. [TRANSITION ACT] put on shield

[TRIPLE] you have pickaxe
[TRIPLE] moss in West Royal Road [TRIPLE] you have King’s Order [TRIPLE] you have gear
[TRIPLE] you have green moss
[TRIPLE] Kraxis in West Royal Road [TRIPLE] you have Sword
[TRIPLE] ground in West Royal Road [TRIPLE] you have shield
[TRIPLE] you in West Royal Road [TRIPLE] you have lantern
[TRIPLE] you have magical torch
[TRANSITION ACT] put on shield

[OBS] Inventory: You are carrying:
- a green moss (providing light)
- a gear
- a pickaxe
- a magical torch (providing light)
- a Sword
- a shield
- King’s Order
- a lantern (providing light)

[ACT] say manaz
[ACT] north
[ACT] west
[ACT] put order down
[ACT] put moss down
[ACT] put lantern down
[ACT] put pickaxe down
[ACT] put shield down
[ACT] put sword down
[ACT] put gear down
[ACT] put all down
[ACT] put on shield
[ACT] wield sword

Worldformer-BART Predictions

<table>
<thead>
<tr>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
</table>

AWM-BART Predictions

<table>
<thead>
<tr>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
</table>

Table 13: Comparison of model predictions on a test set example from Deephome.
### Input to the Model

<table>
<thead>
<tr>
<th>Observation</th>
<th>Graph</th>
<th>Inventory</th>
<th>Valid actions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>[OBS]</strong> Location: Storage</td>
<td><strong>[TRIPLE]</strong> mysterious vial in shelves <strong>[TRIPLE]</strong> you in Storage</td>
<td><strong>[OBS]</strong> Inventory: You are carrying nothing.</td>
<td><strong>[ACT]</strong> take vial <strong>[ACT]</strong> take all off shelves <strong>[ACT]</strong> north</td>
</tr>
<tr>
<td>This is a minor storage chamber, connected to the study through a doorway in the northern wall. There are some shelves, mostly vacant, here. On one of the shelves are four small vials, each neatly labelled and containing some green powder. <strong>[OBS]</strong> Storage</td>
<td><strong>[TRIPLE]</strong> shelves in Storage</td>
<td><strong>[TRANSITION ACT]</strong> take all off shelves</td>
<td></td>
</tr>
<tr>
<td><strong>[TRIPLE]</strong> dark tower in Storage</td>
<td><strong>[TRANSITION ACT]</strong> take all off shelves</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>[TRIPLE]</strong> Storage south Study</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Worldformer-BART Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positives</td>
</tr>
<tr>
<td>'put vial down', 'put all on shelves', 'north'</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AWM/BART Predictions</th>
</tr>
</thead>
<tbody>
<tr>
<td>True Positives</td>
</tr>
<tr>
<td>'put vial down', 'put vial on shelves', 'put all on shelves', 'north'</td>
</tr>
</tbody>
</table>

Table 14: Comparison of model predictions on a test set example from Temple.