DiffG-RL: Leveraging Difference between State and Common Sense

Tsunehiko Tanaka *
Waseda University
tsunehiko@fuji.waseda.jp

Daiki Kimura
IBM Research
daiki@jp.ibm.com

Michiaki Tatsubori
IBM Research
mich@jp.ibm.com

Abstract

Taking into account background knowledge as the context has always been an important part of solving tasks that involve natural language. One representative example of such tasks is text-based games, where players need to make decisions based on both description text previously shown in the game, and their own background knowledge about the language and common sense. In this work, we investigate not simply giving common sense, as can be seen in prior research, but also its effective usage. We assume that a part of the environment states different from common sense should constitute one of the grounds for action selection. We propose a novel agent, *DiffG-RL*, which constructs a Difference Graph that organizes the environment states and common sense by means of interactive objects with a dedicated graph encoder. *DiffG-RL* also contains a framework for extracting the appropriate amount and representation of common sense from the source to support the construction of the graph. We validate *DiffG-RL* in experiments with text-based games that require common sense and show that it outperforms baselines by 17% of scores. We will make our code publicly available.

1 Introduction

Taking into account background knowledge as the context has always been an important yet challenging part of solving tasks that involve natural language. One illustrative example of such challenges is text-based games. Text-based games are computer games where game states and action spaces are represented in pure texts. To play them, players have to not only understand in-game texts correctly but also make appropriate action decisions from given options according to the context. Computational agents required to solve such games naturally arise in the form of natural language processing (NLP) systems trained with reinforcement learning

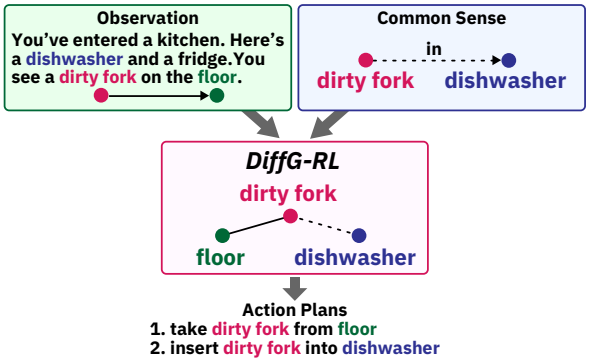

Figure 1: An example of text-based games. Our proposed *DiffG-RL* summarizes the difference between the environment states and common sense in a graph and uses it as a basis for predicting a plan of action.

(RL) algorithms. However, the intrinsic properties of text-based games such as partial observability, long-term dependencies, sparse reward signals, and large action spaces make it extremely challenging for RL agents to learn. Specifically, the chance of agents discovering optimal actions from the vast combinatorial action spaces is astronomically low.

Previous approaches (Ammanabrolu and Riedl, 2019b; Murugesan et al., 2021b; Sahith et al., 2020; Murugesan et al., 2021a; Kimura et al., 2020; Tanaka et al., 2022) have used external knowledge to constraint agents' action outputs in order to shrink the size of search space. Recently, Murugesan et al. (2021a) and Tanaka et al. (2022) utilized human common sense which improved sample efficiency and enabled agents to perform look-ahead planning. However, these approaches have not yet clarified how common sense should actually be used. Specifically, a huge amount of common sense is given at once, regardless of the environment states, and the correspondence between the states and common sense is unclear. This prevents agents from learning which common sense to use in which state, and the agents remember only the results after using common sense.

*Work done during an internship at IBM Research.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 1534–1546
December 7-11, 2022

In this paper, as usage of common sense, we assume that differences between environment states and common sense can provide a basis for action selection and further improve sample efficiency. For example, consider the environment state of "dirty fork on the floor" and the common sense of "dirty fork should be in the dishwasher". The difference between floor and dishwasher in the location of the dirty fork helps agents plan their actions to pick it up from the floor and put it in the dishwasher. We construct a difference graph that maps environment states to common sense and explicitly represents their differences. Further, we develop an encoder dedicated to this graph and propose an agent that can effectively concentrate on learning which common sense to use in which state. An example is shown in Fig. 1.

Inevitable problems in constructing the difference graph are extracting the right amount and unifying the representation. For the first problem, large amounts of common sense cannot be encoded, and small amounts are insufficient for learning. In prior work, the amount of common sense is reduced by extracting only common sense that contains representations that exactly match the objects that appear in the environment, but in some tasks, there may be no common sense available due to mismatches of representations. In contrast, we extract the appropriate amounts of common sense based on semantics and the circumstances, independent of linguistic representations. For the second problem, to help agents recognize the difference between common sense and environmental states, their representations should be aligned. Therefore, we propose an extraction framework for common sense that includes an acquisition of appropriate amounts based on meanings and circumstances and a representation transformation that facilitates the mapping to environment states.

Our contributions in this work are as follows. (1) We introduce a difference graph with an explicit representation of the difference between the environment states and common sense and a novel agent with a dedicated graph encoder. (2) We develop a framework for extracting common sense from sources to facilitate comparing the environment states with common sense. (3) We perform experiments with text-based games that require common sense and demonstrate that our approach outperforms baselines, and evaluate the effect of each component in our approach through ablation studies.

2 Background

Text-based Games: Text-based games can be formally framed as partially observable Markov decision processes (POMDPs), represented as a 7-tuple of $\langle S, T, A, \Omega, O, R, \gamma \rangle$ denoting the set of environment states, conditional transition probabilities, actions, observations, conditional observation probabilities, reward function, and discount factor. We target choice-based games, where the player receives a textual observation $o_t \in \Omega$ and sends a short textual phrase from action choices A to the environment as an action a_t. Most text-based games contain entities ($e_1, e_2, ..., e_M \in E_t$) such as items and location, and players often take actions on themselves ("go east") or on items ("take dirty fork").

Common Sense and Text-based Games: Given common sense, an agent receives an observation o_t to determine the next action by comparing it with the common sense. Common sense is represented by an external knowledge graph stored as triplets of $\langle$subject, relationship, object$\rangle$, which is called a common sense graph. While the recently proposed TWC agent (Murugesan et al., 2021a) uses multiple graphs obtained from ConceptNet (Speer et al., 2017) and combines them, (Tanaka et al., 2022) showed that just a single graph can suffice if Visual Genome (VG) (Krishna et al., 2017) is used, as it contains more grounded graphs. We therefore use VG as a common sense source. There are two challenges when it comes to using VG with common sense: first, how to extract common sense from the source, and second, how agents use the common sense. In this work, we propose two methods to individually address these challenges.

Environment States Extractor: In this study, the current states of the environment are extracted from the observation o_t. Since states in text-based games have a graph-like structure, prior works (Ammanabrolu and Riedl, 2019a; Ammanabrolu and Hausknecht, 2020; Murugesan et al., 2021b; Adhikari et al., 2020) have represented environment states as a dynamic knowledge graph. KG-A2C introduced interactive objects ($io_1, io_2, ..., io_P \in I_t$), which are items that allow agents to interact directly with the surrounding environment. We connect these interactive objects to their state nodes (e.g., the locations they are in) in the knowledge graph and then separate them from the entities ($I_t \subset E_t$).

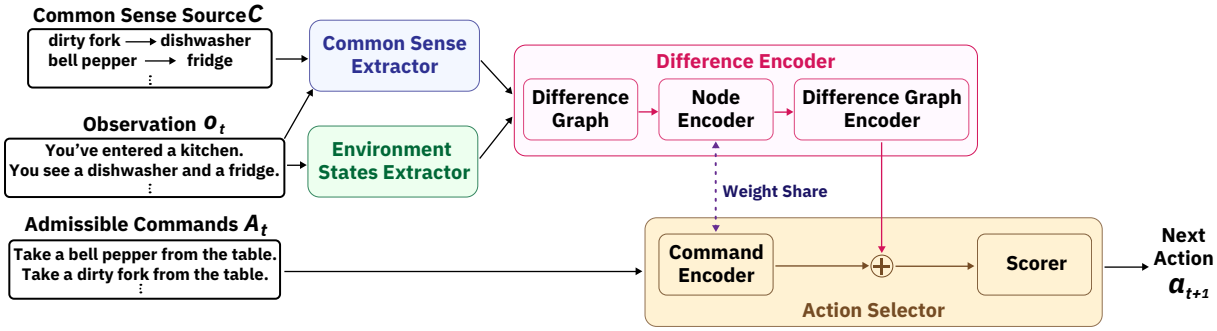

Figure 2: An overview of our proposed *DiffG-RL* for text-based games based on the difference between the environment states and common sense. The Model consists of four components: Environment States Extractor, Common Sense Extractor, Difference Encoder, and Action Selector.

We build on this to extract the states of interactive objects from observation text by using the Abstract Meaning Representation (AMR) parser (Zhou et al., 2021). Nodes connected to the interactive objects by positional relationships ("on," "in," Etc.) from AMR are extracted as the current states. We also add a node representing the player "You" and use it when the interactive objects are in the inventory. At each step, the entities E_t, the interactive objects I_t, and their current states are updated on the basis of the observations, and the states of all interactive objects are tracked.

3 *DiffG-RL*

In this section, we first present an overview of our proposed agent, called *DiffG-RL*, and then propose a framework for the extraction of common sense to facilitate the comparison of differences between environment states and common sense. Finally, we introduce the Difference Graph and its encoder, which provides a representation of the difference.

3.1 Model Overview

Our *DiffG-RL* agent with common sense for text-based games consists of the four components shown in Fig. 2.

1. Environment States Extractor (EE): A component that extracts interactive objects and their current states from the observation texts o_t. The details were described in Section 2.

2. Common Sense Extractor (CE): A framework for extracting the appropriate amount and representation of common sense to facilitate comparisons of the environment states and common sense.

3. Difference Encoder (DE): A graph encoder and a node encoder of a difference graph that organize the environment states and common sense for interactive objects.

4. Action Selector (AS): A component that includes an encoder of admissible commands A_t and a selector that determines the next action a_{t+1} from the output of the difference graph encoder and the command encoder.

3.2 Common Sense Extractor

Giving the common sense all at once (e.g., all the triples of graphs in ConceptNet) is excessive and inefficient for solving the tasks. We have also not identified any way to encode it all at once so far. Existing research (Murugesan et al., 2021a; Tanaka et al., 2022) has narrowed it down to triples of common sense graphs ($c_1, c_2, ..., c_N \in C$) that perfectly match the entities ($e_1, e_2, ..., e_M \in E$) in text-based games, but common sense rarely has such a representation. We extract common sense based on word meaning and the circumstances of games, independent of linguistic representation, to broaden the scope of common sense extraction. In addition, the difference graph described in Section 3.3 needs unified representations of environment states and common sense to help agents understand the correspondences between them. To extract the appropriate amount and representation of common sense, we propose a framework consisting of three components: extracting by meaning (EbM), narrowing by circumstances (NbC), and transforming into grounded representation (TGR).

3.2.1 Extracting by Meaning

As the first step to extract triples of common sense graphs by meaning, we utilize the similarity be-

tween vectors obtained by word embedding instead of spell matching. The similarity sim is represented as

$$sim(s_i, e_j) = \frac{\boldsymbol{s_i} \cdot \boldsymbol{e_j}}{|\boldsymbol{s_i}||\boldsymbol{e_j}|}, \quad (1)$$

where s_i is a subject in a triple of common sense graph c_i, and e_j represents an entity in text-based games. The bolded terms also represent vectors obtained by word embedding. If the similarity is greater than a preset threshold, it is considered to have a similar meaning. We calculate this for all combinations and then replace s_i with an object in the triple of common sense graph and calculate them again. If both s_i and o_i are similar to one of E, its triple of common sense graph c_i is extracted.

Note that because this component relaxes the constraints on common sense much more than with exact matching, the number of extracted triples of common sense graphs will be enormous. [1] In most cases, it is necessary to use it in combination with the twc components introduced in 3.2.2 and 3.2.3.

3.2.2 Narrowing by Circumstances

We leave only triples of graphs that are in line with the circumstances of games, i.e. "interactive object → object's state". In many text-based games, "interactive object → location" (e.g., dirty fork → dishwasher) remains, while "location → location" (agents do not move the dishwasher into the fridge) etc. is removed.

3.2.3 Transforming into Grounded Representation

We transform the subject and object in the extracted triple of common sense graph into the entities to which they correspond in the first component, the EbM (Section 3.2.1). In the case of Eq. 1, s_i is transformed into e_j. This eliminates the influence of differences between the extracted common sense and games' representations and clarifies the correspondence between the environment states and common sense.

3.3 Difference Encoder

3.3.1 Difference Graph

We introduce the difference graph to represent the difference between the environment states and common sense to select common sense according to the states and to obtain the basis for the next action plan, as shown in Fig. 3. We define the difference graph as a representation of the situation where "an interactive object should be placed at A based on common sense, but is now placed at B" (a dirty fork should be placed at the dishwasher but is currently on the floor). The outputs of the current state extractor and the common sense extractor are organized by interactive objects ($io_1, io_2, ..., io_P \in I_t$). The difference graph contains three types of nodes, with multiple current state nodes ($st_1, st_2, ..., st_U \in \mathcal{U}(p)$) and common sense nodes ($co_1, co_2, ..., co_V \in \mathcal{V}(p)$) corresponding to one interactive object node (io_p). For edges, there are two types: interactive object-current state and interactive object-common sense. After the TGR (Section 3.2.3), the common sense node has the same representation as the entities in games. The difference graph is updated in accordance with the observation texts at each time step.

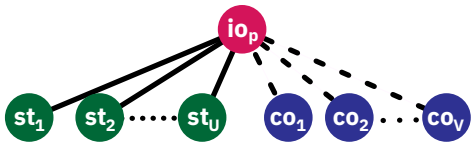

Figure 3: Concept of the difference graph. Pink nodes, green nodes, and blue nodes represent interactive objects, environment states, and common sense, respective.

3.3.2 Node Encoder

We convert the words in a node of the difference graph into a series of vectors by word embedding and obtain a fixed-length vector using the node encoder. We use the fixed-length vector as the initial feature of each node in the difference graph encoder. We use bidirectional GRU (Cho et al., 2014) for the node encoder.

3.3.3 Difference Graph Encoder

We develop a graph encoder to encode the difference graph. Similar to recent graph neural networks, we update the features of a node by aggregating the features of its neighbors. The aggregate of our encoder is based on the Graph Isomorphism Network (GIN) (Keyulu et al., 2019) and is calcu-

[1] We tried doing the extraction with an NVIDIA TITAN X (Pascal) with 12 GB of memory, but the triples extracted from VG overwhelmed the available memory.

lated as

$$h_{io_p}^{(k)} = \text{MLP}\{\phi(1 + W_I)h_{io_p}^{(k-1)} + \phi(W_{ST}h_{st_u}^{(k-1)}) + \phi(W_{CO}h_{co_v}^{(k-1)})\}, \quad (2)$$

where $h_X^{(k)}$ represents the feature of X node with k iterations of the aggregation, ϕ represents an activation function, and MLP represents multi-layer perceptrons.

To distinguish between the three types of node and represent the difference between the current state and common sense, different learnable parameters are provided for each type: W_I, W_{ST}, and W_{CO}. Since the actions are based on the interactive objects, the encoder only aggregates for the interactive object io_p. In GIN, one MLP is used after the product with the learnable parameter because MLPs can represent a composition of functions, but we add an activation function ϕ for output simplification and training stability.

The aggregation can be repeated to reflect the features of distant nodes. As a results, the difference graph encoder can handle the environment states and common sense even if they become subgraphs consisting of multiple nodes. Note that we assume the maximum distance of 1 from the interactive object in the following experiments.

3.4 Action Selector

The action selector calculates the probability of each action from the concatenation of the vector representation $\boldsymbol{a_t^i}$ of the admissible command $a_t^i \in A_t$ and the output $\boldsymbol{d_t}$ of the difference graph encoder. $\boldsymbol{a_t^i}$ is obtained by word embedding and the command encoder, similar to the node encoder in the difference graph encoder. We use the bidirectional GRU for the command encoder and share weight with the node encoder. The scorer consists of two MLP layers, a dropout layer, and an activation layer and calculates the probability $p_{a_t^i} = \text{Scorer}(\boldsymbol{a_t^i}, \boldsymbol{d_t})$.

4 Experiments

4.1 Environment

We conduct experiments with the TWC game (Murugesan et al., 2021a) to verify the difference between common sense and the environment states. The goal of the TWC game is to tidy up a house by putting items where they should be and requires common sense about the relationships between objects and their locations. We generate a new game set using the scripts from the original TWC study (Murugesan et al., 2021a). There are three difficulty levels depending on the number of rooms and the number of interactive objects, as shown in Tab. 1. Because the agent performance is affected by the number of objects/rooms, we unify the different numbers of these included in the same difficulty level in the original dataset.

Table 1: Specifications of TWC game that we use.

Level	Interactive objects	Rooms
Easy	1	1
Medium	3	1
Hard	7	2

To test the generalization performance, we introduce a supervised learning paradigm and split the dataset into three subsets: train, test, and valid. The original dataset (Murugesan et al., 2021a) contains only five games each in the train and test sets, but we generate 100 games and split them into train : test : valid $= 50 : 40 : 10$.

The TWC game contains two test sets: an *IN* set with the same entities as the train set and an *OUT* set consisting of entities that do not appear in the train set. The *OUT* set cannot be solved simply by memorizing the results (i.e., pairs of interactive objects and locations) after using common sense. Depending on the situation, such as the type of room the agents are in now, the same thing may be placed in different places between the train set and the *OUT* set. Therefore, the *OUT* set is used for the validation and ablation study because it is suitable for evaluating the ability to use common sense in a given situation.

4.2 Methods and Metrics

We use four baselines.

- KG-A2C (Ammanabrolu and Hausknecht, 2020) is a method with our implementation that organizes the environment states obtained from observation texts into a knowledge graph.
- TWC agent-CN (Murugesan et al., 2021a) is a method that uses common sense obtained from ConceptNet.
- TWC agent-VG (Tanaka et al., 2022) is a method that utilizes the same model as

Table 2: General results for two test games: *IN* within the training distribution of entities and *OUT* outside the distribution. All experiments were performed with five random seeds. Each value is a pair (average) ± (standard deviation). We highlight the **best** model in bold.

	Method	Easy	Medium	Hard
IN	KG-A2C (Ammanabrolu and Hausknecht, 2020)	0.89 ± 0.02	0.76 ± 0.02	0.33 ± 0.01
	TWC agent-CN (Murugesan et al., 2021a)	0.91 ± 0.02	0.75 ± 0.02	0.31 ± 0.01
	TWC agent-VG (Tanaka et al., 2022)	0.92 ± 0.01	0.69 ± 0.03	0.32 ± 0.02
	TWC agent-VG+KG-A2C	**0.95 ± 0.01**	**0.82 ± 0.02**	0.26 ± 0.01
	DiffG-RL	**0.95 ± 0.00**	**0.82 ± 0.02**	**0.38 ± 0.02**
OUT	KG-A2C (Ammanabrolu and Hausknecht, 2020)	0.78 ± 0.03	0.72 ± 0.02	0.33 ± 0.01
	TWC agent-CN (Murugesan et al., 2021a)	0.77 ± 0.03	0.69 ± 0.02	0.29 ± 0.02
	TWC agent-VG (Tanaka et al., 2022)	0.78 ± 0.03	0.67 ± 0.02	0.25 ± 0.02
	TWC agent-VG+KG-A2C	0.82 ± 0.03	0.72 ± 0.02	0.25 ± 0.01
	DiffG-RL	**0.91 ± 0.04**	**0.76 ± 0.02**	**0.35 ± 0.02**

TWC agent-CN but with VG as the common sense source.

- TWC agent-VG+KG-A2C is a method that naively combines the TWC agent-VG and KG-A2C without our proposed difference encoder and common sense extractor.

We evaluate performance on the normalized score computed by dividing the actual score by the maximum possible score. The scores range from 0 to 1, and higher is better.

4.3 Implementation and Training Details

In all methods, we use GloVe (Pennington et al., 2014) provided by GENSIM [2] for word embedding. The hidden size is set to 512 for *DiffG-RL* and to 300 for the baselines. The activation function is ELU (Clevert et al., 2016) for *DiffG-RL* and ReLU (Nair and Hinton, 2010) for the baselines. The threshold of similarity for the EbM (Section 3.2.1) is set to 0.3.

For training, we optimize all models for 100 epochs with Adam (Kingma and Ba, 2015) optimizer using a learning rate of 3.0×10^{-5} and the default hyperparameters in PyTorch (Paszke et al., 2017). For tests, the model with the largest normalized score and the smallest number of steps in the validation is used. *DiffG-RL* can train 100 epochs in two and a half days using a single NVIDIA TITAN X (Pascal) GPU.

4.4 General Results

Similarity Threshold: Table 2 lists the results of the *IN* and *OUT* test sets achieved by the baselines and the proposed approach (*DiffG-RL*) trained for each difficulty level. *DiffG-RL* using the difference graph outperforms the baselines in all results. Specifically, *DiffG-RL* improves 40% in the hard level using the *OUT* set from TWC agent-VG, which uses only common sense, and 17% in the easy level using the *OUT* set from KG-A2C (previous SOTA), which uses only a knowledge graph. We can observe high performances on *OUT*, which cannot be solved by simply memorizing the results after using common sense in the training. This suggests that the representation of the difference between the environment states and common sense in the proposed approach contributes to learning how to use common sense.

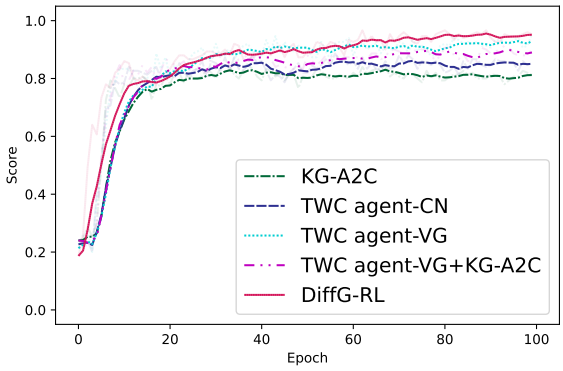

Figure 4: Performance evaluation for the medium level in the training. (Smoothing is performed to clarify the differences in the results of a single run.)

Table 2 also shows that TWC agent-VG+KG-A2C struggles on the hard difficulty level and is less than or equal to KG-A2C and TWC agent-VG, which use only a knowledge graph of the environment states or common sense (not both). We believe this approach is vulnerable to an increase in the environment states and common sense as

[2] https://radimrehurek.com/gensim/

Table 3: Comparison of performance to evaluate the components, EbM, NbC, TGR (Section 3.2), and DE (Section 3.3) in *DiffG-RL*. We use hard level games in *OUT*.

No.	EbM	NbC	TGR	DE	Precision	Recall	Scores
1					23 / 896 (2.6%)	23 / 257 (6.4%)	0.25 ± 0.02
2	✓				2.80M / 11.00M (25.6%)	343 / 357 (96.1%)	N/A
3	✓	✓			2.80M / 6.05M (37.1%)	343 / 357 (100%)	N/A
4	✓	✓	✓		343 / 5414 (6.3%)	343 / 357 (96.1%)	0.26 ± 0.01
5	✓	✓	✓	✓	343 / 5414 (6.3%)	343 / 357 (96.1%)	$\mathbf{0.35 \pm 0.02}$

Table 4: Ablation results for the relationship between the similarity threshold (TH) in EbM and the extracted common sense graphs. CE denotes the common sense extractor (Section 3.2). We use goal graphs of hard level games in the *OUT* set.

No.	CE	TH	Precision	Recall
1		-	23 / 896 (2.6%)	23 / 357 (6.4%)
2	✓	0.6	268 / 3707 (7.2%)	268 / 357 (75.1%)
3	✓	0.5	321 / 2984 (10.8%)	321 / 357 (89.9%)
4	✓	0.4	341 / 5151 (6.6%)	341 / 357 (95.5%)
5	✓	0.3	343 / 5414 (6.3%)	343 / 357 (96.1%)

the number of interactive object increases. In contrast, *DiffG-RL* shows a solid improvement over KG-A2C and TWC agent-VG, indicating that it is able to effectively utilize a combination of the environment states and common sense to deal with increased interactive objects.

Figure 4 shows the training curves at the medium level for the baselines and *DiffG-RL*, where it is clear that *DiffG-RL* performs the best. We believe the difference between the environment states and common sense has a positive impact on the decision making and improves the sample efficiency.

Considering both Tab. 2 and Fig. 4 together, in the medium level games, we can see that TWC agent-VG has a low performance on both test sets, despite its high performance in the training. In contrast, *DiffG-RL* performs well on all of the training and the two test sets. This reinforces our intuition that the difference graph of our proposed approach improves the generalization performance.

4.5 Ablation Study

Effect of Components: We investigate the effect of the components in *DiffG-RL*, and show the results in Tab. 3. Based on the field of information retrieval, we add precision and recall to the metric to evaluate how well the extraction method can extract the common sense needed to achieve the game goals. We first introduce the concept of a goal graph $g_1, g_2, ..., gL \in G$ directly connecting interactive objects and their goal locations. The precision is calculated by dividing the number of triples of common sense graphs that correspond to the triple of goal graph $C_g \subset C$ by the total number N of C. The recall is calculated by dividing the number of triples of goal graphs that are covered by the triples of common sense graphs $G_c \subset G$ by the total number L of G.

We believe that an agent's performance can be improved by giving it a computationally feasible number of commonsense knowledge triples, which cover the common sense needed to solve the problem. Thus, higher both precision and recall are better, but there is a tradeoff between the two. We also use TWC agent-VG+KG-A2C as the most naive method (no.1), and since the performance difference between the proposed method and no.1 is the largest in Tab. 2, we use the hard level games in the *OUT* set for the score.

We can see that using EbM significantly improves the recall. This reinforces the effectiveness of our idea that common sense should be extracted by meaning independent of linguistic representation. We also see that the precision is also greatly improved by using NbC together. However, the number of triples of common sense graphs extracted is still huge, as indicated by the denominator values of the precision in no.3. No.2 and no.3 could not be executed because the number of triples exceeds the available GPU memory. Since TGR unified the representation of goal graphs and the extracted common sense graphs, multiple triples can be combined into one and the number of the extracted triples can be reduced. Therefore, we believe that the proposed components should not be used individually but as a framework that brings them together.

We also see from the difference in scores that both CE and DE contribute to the performance improvement. Comparing their respective contributions, we can see that DE has a greater impact on performance than the other components ($4 \rightarrow 5$).

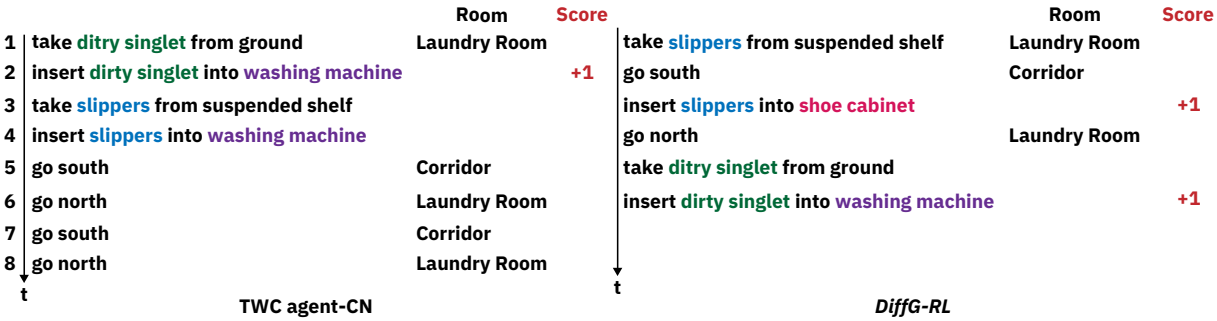

Figure 5: An comparison of agents' behavior. TWC agent-CN (left) and *DiffG-RL* (right) performed the task of placing a dirty singlet and slippers, which is in the hard difficulty level and uses the *OUT* set.

This leads us to believe that the difference graph is relatively critical in *DiffG-RL* for its impact on the agents' decision-making.

We investigate the relationship between the similarity threshold in EbM and the number of the extracted triples of common sense graphs. We compute the precision and the recall between the extracted common sense graphs and the goal graphs in the same way as in Tab. 3 and show the results in Tab. 4. We can observe that the precision and the recall are a tradeoff tendency when the threshold exceeds 0.5. To maximize the performance of DE by providing common sense graphs that correspond to the goal graphs in all games, we focus on the recall in our experiments and set the threshold at 0.3. Note that lowering the threshold increases the noise (the precision decreases), so a method to increase the recall while keeping the threshold high will be required in the future (see Section 7).

4.6 Qualitative Results

Figure 5 shows a comparison of the agent's behavior between TWC agent-CN and *DiffG-RL*. The tasks are to put the dirty singlet in the laundry room (north) ground into the washing machine and to put the slippers in the suspended shelf in the laundry room into the shoe cabinet in the corridor (south).

TWC agent-CN puts the dirty singlet into the washing machine correctly, but it also puts the slippers into the washing machine wrongly. We assume that this is because by giving the agent a single vector that aggregates common sense knowledge triples about all interactive objects, the agent has been strongly affected by common sense for the dirty singlet. We also observe that for slippers where the goal exists in another room, it repeats the move commands (such as "go south" and "go north") and does not reach the correct location. This can be considered an incorrect understanding of the current state. However, *DiffG-RL* can put the three interactive objects back where they should be in order, even if it requires moving the room. We believe *DiffG-RL* is robust to such effects of common sense about other interactive objects because it explicitly encodes the correspondences between the current location and common sense for each interactive object using a difference graph.

5 Related Work

Common Sense for Text-based Games: Many recent methods have focused on providing common sense to agents to efficiently explore the vast observation and action spaces of text-based games. Murugesan et al. (2021a) proposed a text-based game TextWorld Commonsense (described as the TWC game in this paper) that requires common sense from agents and a baseline TWC agent that utilize common sense obtained from ConceptNet (Speer et al., 2017). Tanaka et al. (2022) proposed using scene graph datasets such as VG (Krishna et al., 2017) as a more grounded common sense source base on the TWC agent. BiKE (Murugesan et al., 2021b) shares information between the state graph and the common sense graph by means of a bidirectional attention mechanism, but focuses only on information that is similar between nodes. Ammanabrolu and Riedl (2019b) transfer common sense trained in other games into the target game strategy. Sahith et al. (2020) utilize common sense obtained in large-scale models such as COMET (Bosselut et al., 2019) and BERT (Devlin et al., 2019) based on KG-A2C (Ammanabrolu and Hausknecht, 2020). Common sense also has a high affinity with logic rules, and some studies (Kimura et al., 2020, 2021a,b; Chaudhury et al., 2021) combine them to solve text-based games. Since these methods do not clarify the correspondence between

the environment states and common sense, it is difficult to learn how to use common sense according to the situation.

In contrast, our work extracts common sense by meaning and graphically represents the difference between common sense and the environment states.

Knowledge Graph Extraction: Some prior works have utilized state representations using knowledge graphs to effectively prune the vast observation and action space of text-based games. KG-DQN (Ammanabrolu and Riedl, 2019a) and KG-A2C use OpenIE (Angeli et al., 2015) to create a knowledge graph of the game's belief state from observation texts. GATA (Adhikari et al., 2020) uses self-supervised learning to train the construction and update the belief graph. Worldfromer (Ammanabrolu and Riedl, 2021) uses the world models to simultaneously tasks agents with a next-step belief graph prediction as well as the usual action generation. These methods utilize knowledge graph representation of the environment states, but do not use prior knowledge such as common sense.

Our work differs in that we utilize a knowledge graph to represent not only the environment states but also the correspondence with common sense in the form of the different graphs.

Usage of Common Sense Extraction of suitable commonsense statements is well-studied (Ma et al., 2019; Lin et al., 2019), and recall of commonsense knowledge for a task receives much attention (Lin et al., 2019; Ilievski et al., 2020) in QA. However, as mentioned in 1, we focus on the denominator of precision as well as recall, which we believe is a new perspective. In addition, existing extraction methods (Ma et al., 2019; Lin et al., 2019) use triples with matched words. However, in this study, we use triples that are close in the distance between vectors after transformation by word embedding.

The filtering of common sense statements based on common sense is also studied in works related to the affordance of objects or defeasible reasoning (Qasemi et al., 2022; Rudinger et al., 2020; Do and Pavlick, 2021). They argue that the performance of state-of-the-art language models drops significantly in updating inferences when the context changes from the general situations by using their original tasks and datasets. In contrast, this paper proposes a model that explicitly represents the dynamically changing context of environmental states and common sense differences in RL and argues for its effectiveness through experiments.

6 Conclusion

In this work, we investigated the difference between the environment state and common sense as a basis for RL agent decision-making in text-based games. We proposed *DiffG-RL*, a novel agent that constructs a graph that represents the difference, along with a dedicated encoder, also contains a common sense extraction framework to obtain the appropriate amount and representation of common sense to facilitate the comparison between the environment states and the common sense. Our experimental results showed that *DiffG-RL* outperformed baselines that used only a knowledge graph, only common sense, or a naive combination of the two. These findings demonstrate the effectiveness of the difference graph, which is a representation of the difference between the environment states and common sense, for text-based games.

7 Limitations

An important aspect of our approach is that it utilizes the difference between the environment states and common sense as the basis for decision-making. However, the TWC game does not consider the relationships with other objects, which means the agents were sometimes not provided sufficient context. For example, it is difficult for an agent to determine whether a dirty fork should be placed in a dishwasher or on a dining table based solely on the information that it is holding a dirty fork in its hand. The location depends on further contexts, such as whether there is food left on the plates or whether the person eating is full. Since agents can only repeat their attempts based on scores in the TWC game, we hope to validate agents in games that provide more context, such as ALFWorld (Shridhar et al., 2021) and ScienceWorld (Wang et al., 2022). Our approach should be able to support such high-context situations if we extend the representation of the environment states in the difference graph from node to sub-graph (details are provided in Section 3.3.3).

In addition, although we used GloVe for word embedding, this is slightly outdated considering the recent development of natural language processing. The performance of our approach could be further improved by using more up-to-date word embedding. This may also allow us to obtain a sufficient number of graphs even if we raise the similarity threshold, which was set to 0.3 in our experiments.

8 Broader Impact

Our model does not use sensitive contexts such as legal or medical data. In addition, the dataset and common sense sources used in our experiments do not contain sensitive information. Since the actions taken by agents in the proposed model are based on the difference between the environment states and common sense, we can analyze the difference to reveal the reasons behind the actions. For example, if the model is biased, it can help to identify the cause of the biased behavior. However, when adding new common sense, it is necessary to thoroughly examine the model for bias, including that which has already been added.

References

Ashutosh Adhikari, Xingdi (Eric) Yuan, Marc-Alexandre Côté, Mikulas Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L. Hamilton . 2020. Learning dynamic belief graphs to generalize on text-based games. In *NeurIPS 2020*. ACM.

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. In *International Conference on Learning Representations*.

Prithviraj Ammanabrolu and Mark Riedl. 2019a. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3557–3565, Minneapolis, Minnesota. Association for Computational Linguistics.

Prithviraj Ammanabrolu and Mark Riedl. 2019b. Transfer in deep reinforcement learning using knowledge graphs. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 1–10, Hong Kong. Association for Computational Linguistics.

Prithviraj Ammanabrolu and Mark Riedl. 2021. Learning knowledge graph-based world models of textual environments. In *Advances in Neural Information Processing Systems*.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. 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)*, pages 344–354, Beijing, China. Association for Computational Linguistics.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Subhajit Chaudhury, Prithviraj Sen, Masaki Ono, Daiki Kimura, Michiaki Tatsubori, and Asim Munawar. 2021. Neuro-symbolic approaches for text-based policy learning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3073–3078, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Nam Do and Ellie Pavlick. 2021. Are rotten apples edible? challenging commonsense inference ability with exceptions. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2061–2073, Online. Association for Computational Linguistics.

Filip Ilievski, Pedro Szekely, Jingwei Cheng, Fu Zhang, and Ehsan Qasemi. 2020. Consolidating commonsense knowledge. *arXiv preprint arXiv:2006.06114*.

Xu Keyulu, Hu Weihua, Leskovec Jure, and Jegelka Stefanie. 2019. How powerful are graph neural networks? In *International Conference on Learning Representations*.

Daiki Kimura, Subhajit Chaudhury, Masaki Ono, Michiaki Tatsubori, Don Joven Agravante, Asim Munawar, Akifumi Wachi, Ryosuke Kohita, and Alexander Gray. 2021a. LOA: Logical optimal actions for text-based interaction games. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System*

Demonstrations, pages 227–231, Online. Association for Computational Linguistics.

Daiki Kimura, Subhajit Chaudhury, Akifumi Wachi, Ryosuke Kohita, Asim Munawar, Michiaki Tatsubori, and Alexander Gray. 2020. Reinforcement learning with external knowledge by using logical neural networks. In *IJCAI-PRICAI-W*.

Daiki Kimura, Masaki Ono, Subhajit Chaudhury, Ryosuke Kohita, Akifumi Wachi, Don Joven Agravante, Michiaki Tatsubori, Asim Munawar, and Alexander Gray. 2021b. Neuro-symbolic reinforcement learning with first-order logic. In *Empirical Methods in Natural Language Processing*, pages 3505–3511. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2829–2839, Hong Kong, China. Association for Computational Linguistics.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32, Hong Kong, China. Association for Computational Linguistics.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Pushkar Shukla, Sadhana Kumaravel, Gerald Tesauro, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2021a. Text-based RL Agents with Commonsense Knowledge: New Challenges, Environments and Baselines. In *Thirty Fifth AAAI Conference on Artificial Intelligence*.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2021b. Efficient text-based reinforcement learning by jointly leveraging state and commonsense graph representations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 719–725, Online. Association for Computational Linguistics.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Ehsan Qasemi, F LIU, Muhao Chen, and Pedro Szekely. 2022. Paco: Preconditions attributed to commonsense knowledge. EMNLP.

Rachel Rudinger, Vered Shwartz, Jena D. Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A. Smith, and Yejin Choi. 2020. Thinking like a skeptic: Defeasible inference in natural language. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4661–4675, Online. Association for Computational Linguistics.

Dambekodi Sahith, Frazier Spencer, Ammanabrolu Prithviraj, and Mark O. Riedl. 2020. Playing text-based games with common sense. In *Advances in Neural Information Processing Systems Workshops*.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4444–4451. AAAI Press.

Tsunehiko Tanaka, Daiki Kimura, and Michiaki Tatsubori. 2022. Commonsense Knowledge from Scene Graphs for Textual Environments. In *Thirty Fifth AAAI Conference on Artificial Intelligence Workshop (AAAIW)*.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader?

Jiawei Zhou, Tahira Naseem, Ramón Fernandez Astudillo, Young-Suk Lee, Radu Florian, and Salim Roukos. 2021. Structure-aware fine-tuning of

sequence-to-sequence transformers for transition-based AMR parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6279–6290, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Appendix

A.1 Additional Results

Hidden Size: We investigate the optimal hidden size for *DiffG-RL* by training five agents with different hidden sizes at the medium level and testing them in the *OUT* set. The results are shown in Fig. 5. Since we could not find a consistent trend, we used 512 in this work, which had the best performance throughout the experiments in this work.

Table 5: Ablation results to evaluate the hidden size of *DiffG-RL*. All results are in the medium difficulty level using the *OUT* set.

Hidden size	Scores
128	0.71 ± 0.02
256	0.70 ± 0.02
300	0.67 ± 0.02
512	$\mathbf{0.76 \pm 0.02}$
1024	0.68 ± 0.03

Activation Function in Difference Graph Encoder: We investigate the effectiveness of the activation function in our proposed difference graph encoder (Section 3.3.3) added from the base GIN (Keyulu et al., 2019). Table 6 shows that the performance is better with the activation function than without.

Table 6: Ablation results to evaluate the activation function in the difference graph encoder (Section 3.3.3). All results are in the medium difficulty using the *OUT* set.

Method	Scores
DiffG-RL (w/o activation)	0.64 ± 0.01
DiffG-RL	$\mathbf{0.76 \pm 0.02}$

Common Sense Source: We investigate common sense sources. Table 7 shows that in the hard difficulty level using the *OUT* set, TWC agent with ConceptNet performs better than that with VG, but *DiffG-RL* does the opposite. Tanaka et al. (2022) propose that Visual Genome contains more grounded common sense, but TWC agent shows that it does not have the structure to take advantage of it. In contrast, *DiffG-RL* is able to fully utilize VG. Based on this result, we use VG in our experiments.

Table 7: Ablation results to evaluate common sense sources. All results are in the hard difficulty using the *OUT* set. CN and VG denote ConceptNet (Speer et al., 2017) and Visual Genome (Krishna et al., 2017), respectively.

Method	Source	Scores
TWC agent-CN	CN	0.29 ± 0.02
TWC agent-VG	VG	0.25 ± 0.02
DiffG-RL-CN	CN	0.31 ± 0.02
DiffG-RL-VG	VG	$\mathbf{0.35 \pm 0.02}$