Deep Reinforcement Learning with Stacked Hierarchical Attention for Text-based Games

Yunqiu Xu[*]
University of Technology Sydney
Yunqiu.Xu@student.uts.edu.au

Meng Fang[*]
Tencent Robotics X
mfang@tencent.com

Ling Chen
University of Technology Sydney
Ling.Chen@uts.edu.au

Yali Du
University College London
yali.du@ucl.ac.uk

Joey Tianyi Zhou
IHPC A*STAR
zhouty@ihpc.a-star.edu.sg

Chengqi Zhang
University of Technology Sydney
Chengqi.Zhang@uts.edu.au

Abstract

We study reinforcement learning (RL) for text-based games, which are interactive simulations in the context of natural language. While different methods have been developed to represent the environment information and language actions, existing RL agents are not empowered with any reasoning capabilities to deal with textual games. In this work, we aim to conduct explicit reasoning with knowledge graphs for decision making, so that the actions of an agent are generated and supported by an interpretable inference procedure. We propose a stacked hierarchical attention mechanism to construct an explicit representation of the reasoning process by exploiting the structure of the knowledge graph. We extensively evaluate our method on a number of man-made benchmark games, and the experimental results demonstrate that our method performs better than existing text-based agents.

1 Introduction

Language plays a core role in human intelligence and cognition [14, 43]. Text-based games [13, 20], where both the states and actions are described by textual descriptions, are suitable simulation environments for studying the language-informed decision making process. These games can be regarded as an intersection of natural language processing (NLP) and reinforcement learning (RL) tasks [35]. To solve text-based games via RL, the agent has to tackle many challenges, such as learning representation from text [42], making decisions based on partial observations [4], handling combinatorial action space [57] and sparse rewards [56]. Generally, existing agents for text-based games can be classified as rule-based agents and learning-based agents. Rule-based agents, such as NAIL [21], solve the games based on pre-defined rules, engineering tricks, and pre-trained language models. By heavily relying on prior knowledge of the games, these agents lack flexibility and adaptability. With the progress of deep reinforcement learning [38, 39], learning-based agents such as LSTM-DRQN [42] become increasingly popular since they learn purely from interaction without requiring expensive human knowledge as prior. Recently, considering the rich information that can be maintained by its structural memory, knowledge graphs (KGs) have been incorporated into RL agents to facilitate solving text-based games [1, 4, 3].

[*]Equal contribution.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

While a lot of studies have been conducted on representing useful information from text observations [3, 4, 42] and reducing action spaces [20, 57], few RL agent addresses the reasoning process for text-based games. Going beyond mapping a question to an answer, human beings have the ability of reasoning − they can reuse the knowledge [50], or compose the supporting facts (e.g., the relation between objects in the scene) from the question and the knowledge base to interpret the answer [10, 30]. We believe that RL agents empowered with reasoning capabilities will be better mimicking human decisions in solving text-based games and achieving enhanced performance. In terms of RL agents, we consider enhancing the reasoning capability of the agent by exploiting KGs. While existing studies [3, 4, 58] treat KGs as a part of the observation to handle partial observability, they ignore the potential of KGs for reasoning [12, 27]. Furthermore, the effectiveness of reasoning is constrained by two problems. Firstly, existing KG-based agents construct one single KG, so that fine-grained information (e.g., the types of object relationship, the newness/oldness of information) is hard to be maintained. Secondly, the multi-modal inputs, such as textual observations and KGs, are aggregated via simple concatenation so that their respective benefits cannot be sufficiently exploited.

We believe that an intelligent agent should have the ability to conduct explicit reasoning with relational and temporal awareness being taken into consideration to make decisions. In this paper, our goal is to design an enhanced RL agent with a reasoning process for text-based games. We propose a new method, named as **S**tacked **H**ierarchical **A**ttention with **K**nowledge **G**raphs (SHA-KG)[2], to enable the agent to perform multi-step reasoning via a hierarchical architecture on playing games. Briefly, to leverage the structure information of a KG that maintains the agent's knowledge about the game environment, we first consider the sub-graphs of the KG with different semantic meanings so that relational and temporal awareness will be taken into account. Secondly, a stacked hierarchical attention module is devised to build effective state representation from multi-modal inputs, so that their respective importance will be considered.

Our contributions include four aspects. Firstly, our work is a first step in pursuing reasoning in solving text-based games. Secondly, we propose to incorporate sub-graphs of the KG into decision making to introduce the reasoning process. Thirdly, we propose a new stacked hierarchical attention mechanism for RL approach featured by multi-level and multi-modal reasoning. Fourthly, we extensively evaluate our method on a wide range of text-based benchmark games, achieving favorable results compared with the state-of-the-art methods.

2 Related work

Agents for text-based games. Existing agents either perform based on predefined rules or learn to make responses by interacting with the environment. Rule-based agents [8, 16, 21, 31] attempt to solve text-based games by injecting heuristics. They are thus not flexible since a huge amount of prior knowledge is required to design rules [20]. Learning-based agents [2, 20, 22, 26, 42, 55, 56, 57] usually employ deep reinforcement learning algorithms to deliver adaptive game solving strategies. However, the performance of these agents is still not up to par when playing complex man-made games, even though efforts have been made to reduce the difficulty (e.g., DRRN [22] assumed that an action can only be selected from a valid action set for each state). KG-based agents have been developed to enhance the performance of learning-based agents with the assistance of KGs. KGs can be constructed by simple rules so that it substantially reduces the amount of prior knowledge required by rule-based agents. While KGs have been leveraged to handle partial observability [3, 4, 58], reduce action space [3, 4], and improve generalizability [1, 5], few of the existing works addresses its potential for reasoning. Recently, Murugesan et al. [41] tried to introduce commonsense reasoning for playing synthetic games. They extracted sub-graphs from ConceptNet [44], which is a large-scale external knowledge base with millions of edges and nodes. In contrast, we aim to construct the KG based on domain information with minimal external knowledge. Besides, we focus on man-made games which are more complex than synthetic games in terms of logic, so that the reasoning ability becomes especially crucial and desirable to the agents.

Attention mechanism. Attention mechanism has been widely studied in areas of machine learning, psychology and neuroscience [33]. For text-based games, self-attention [45] has been applied to encode textual observation [1, 58], and Graph Attention Networks (GATs) [46] has been employed to encode KGs [3]. Regarding model explainability, the attention mechanism helps to solve the outcome

[2]Our code is available at `https://github.com/YunqiuXu/SHA-KG`.

explanation problem, e.g. building an attention-based saliency map [19]. For RL, the attention mechanism has been used to interpret the decision making process, mostly in tasks with visual inputs [18, 40]. For tasks with multi-modal inputs, such as visual question answering (VQA) [7], the attention mechanism has been used to aggregate the image and text inputs [23, 24, 29, 30, 34, 36]. In this work, we apply an attention mechanism to consider the multi-modal inputs based on text observations and graph structures.

Reasoning via knowledge graph. A lot of existing studies have used KGs as the knowledge base to facilitate learning and interpretation, including incorporating KG-based commonsense reasoning for question answering [9, 15, 32, 61] and recommendation [47, 48, 52, 60]. We are the first to exploit KGs to induce reasoning for RL-based agents playing text-based games.

3 Preliminaries

POMDP Partially Observable Markov Decision Processes (POMDPs) can be defined as a 7-tuple: the state set $\boldsymbol{\mathcal{S}}$, the action set $\boldsymbol{\mathcal{A}}$, the state transition probabilities $\boldsymbol{T}$, the reward function $\boldsymbol{R}$, the observation set $\boldsymbol{\Omega}$, the conditional observation probabilities $\boldsymbol{O}$ and the discount factor $\boldsymbol{\gamma} \in (0, 1]$. At each time step, the agent will receive an observation $\boldsymbol{o}_t \in \boldsymbol{\Omega}$, depending on the current state and previous action via the conditional observation probability $O(\boldsymbol{o}_t|\boldsymbol{s}_t, \boldsymbol{a}_{t-1})$. By executing an action $\boldsymbol{a}_t \in \boldsymbol{\mathcal{A}}$, the environment will transit into a new state based on the state transition probability $T(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$, and the agent will receive the reward $\boldsymbol{r}_{t+1} = R(\boldsymbol{s}_t, \boldsymbol{a}_t)$. Same as Markov Decision Process (MDPs), the goal of the agent is to learn an optimal policy $\boldsymbol{\pi}^*$ to maximize the expected future discounted sum of rewards from each time step: $\boldsymbol{R}_t = \mathbb{E}[\sum_{k=0}^{\infty} \boldsymbol{\gamma}^k \boldsymbol{r}_{t+k+1}]$.

KG Knowledge Graph (KG) for a text-based game can be built from a set of triplets $\langle Subject, Relation, Object \rangle$, denoting that the $Subject$ has $Relation$ with the $Object$. For example, $\langle Kitchen, Has, Food \rangle$. The KG is denoted as $G = (V, E)$, where V and E are the node set and the edge set, respectively. Both $Subject$ and $Object$ belong to the node set V. $Relation$, which corresponds to the edge connecting them, belongs to E.

4 Methodology

4.1 Problem statement

In this work, we focus on man-made games, which are initially designed for human players [20]. These games are devised with more complex logic and much larger action space than synthetic games [13]. Text-based games require an agent to make automatic responses to achieve specific goals (e.g., escaping from the dungeon) based on received textual information. Raw textual observation contains only the feedback of taking an action (e.g., "Taken" is a textual observation after executing the action "take egg"). As underlying states can not be directly observed by the agent, the text-based games can be formulated as POMDPs. Similar to [3], at every step we construct an input $\boldsymbol{s}_t$ as the combination of three components: a textual observation $\boldsymbol{o}_{t,\text{text}}$, a collected raw score $\boldsymbol{o}_{t,\text{score}}$, and a KG $\boldsymbol{o}_{t,\text{KG}}$ (note that here $\boldsymbol{s}_t$ should not be regarded as a true game state as the games are not fully observable). $\boldsymbol{o}_{t,\text{text}}$ further includes the current state $\boldsymbol{o}_{t,\text{desc}}$ (describing the environment), inventory $\boldsymbol{o}_{t,\text{inv}}$ (describing items collected by a player), game feedback $\boldsymbol{o}_{t,\text{feed}}$, and previous action taken $\boldsymbol{a}_{t-1}$. Fig. 1 (a) shows an example of $\boldsymbol{o}_{t,\text{text}}$.

While $\boldsymbol{o}_{t,\text{text}}$ and $\boldsymbol{o}_{t,\text{score}}$ mainly reflect the current observation, $\boldsymbol{o}_{t,\text{KG}}$ records the game history. Therefore, the KG can help the agent to handle partial observability. At each time step, the triples extracted from the current textual observation $\boldsymbol{o}_{t,\text{text}}$ are used to update the KG as,

$$\boldsymbol{o}_{t,\text{KG}} = \text{GraphUpdate}(\boldsymbol{o}_{t-1,\text{KG}}, \boldsymbol{o}_{t,\text{text}}) \tag{1}$$

Fig. 1 (b) shows an example of $\boldsymbol{o}_{t,\text{KG}}$ and how it updates. We provide details of constructing and updating the KG in Sec. 5.2.

4.2 Sub-graph division

As discussed, existing KG-based agents build only one knowledge graph [1, 3, 4, 58]. To introduce relational-awareness and temporal-awareness, in this work, we divide our KG as multiple sub-graphs.

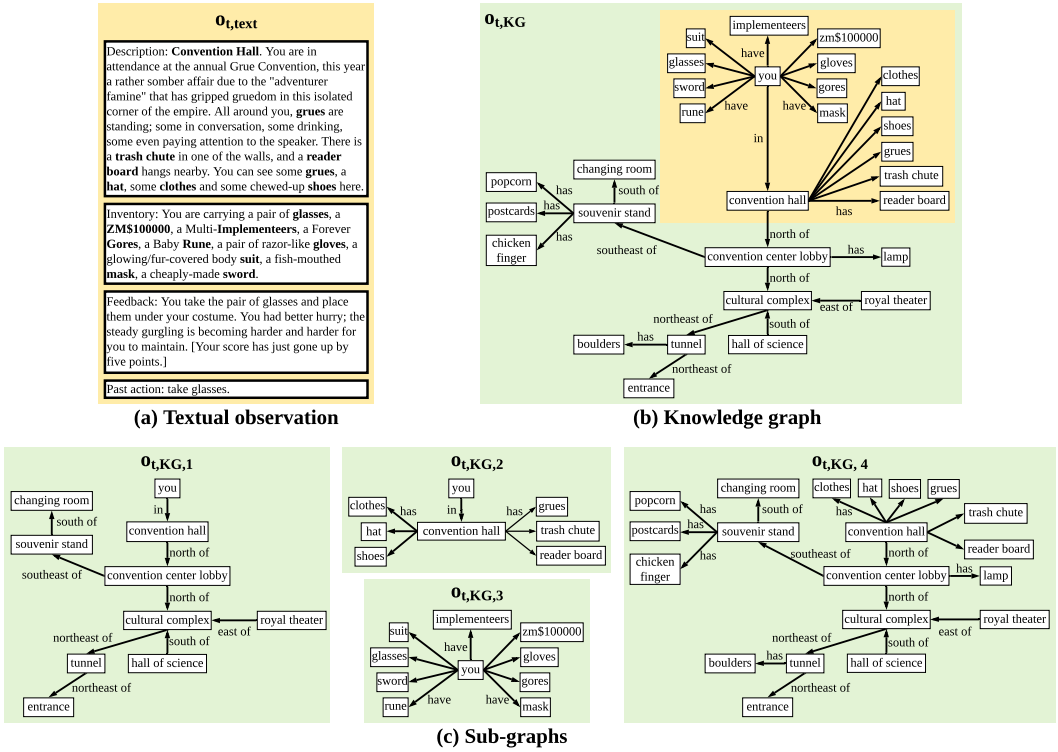

Figure 1: (a) Textual observation $\boldsymbol{o}_{t,\text{text}}$. (b) Knowledge graph $\boldsymbol{o}_{t,\text{KG}}$. Yellow region in $\boldsymbol{o}_{t,\text{KG}}$ contains information extracted from the observation $\boldsymbol{o}_{t,\text{text}}$. (c) Sub-graphs obtained from $\boldsymbol{o}_{t,\text{KG}}$.

Inspired by the heterogeneous graph [49, 59] where a graph contains different types of nodes and edges, we first classify edges by their types (e.g., "Has" and "$East\ of$" can be regarded as different types), and then build relational-aware sub-graphs based on the edges. In addition, as the KG can not distinguish between the current and past information, we introduce temporal-awareness via building different sub-graphs based on whether the historical information is included (e.g., sub-graphs built from $\boldsymbol{o}_{t,\text{text}}$ only and sub-graphs built from $\boldsymbol{o}_{t,\text{text}}$ and $\boldsymbol{o}_{t-1,\text{KG}}$). The union of all sub-graphs is the full KG:

$$\boldsymbol{o}_{t,\text{KG}} = \boldsymbol{o}_{t,\text{KG},1} \cup \boldsymbol{o}_{t,\text{KG},2} \ \ldots\ \cup \boldsymbol{o}_{t,\text{KG,m-1}} \cup \boldsymbol{o}_{t,\text{KG,m}} \tag{2}$$

where m denotes the number of sub-graphs. From the perspective of hierarchical learning [54, 62], the sub-graph division allows observations to be considered in two levels: In the high level, the full KG captures the overall node connectivity. In the low level, the sub-graphs reflect different relational and temporal relations. Fig. 1 (c) shows an example of the sub-graphs obtained from $\boldsymbol{o}_{t,\text{KG}}$.

4.3 Stacked hierarchical attention network

Before action selection, we represent the input $\boldsymbol{s}_t$ as a vector $\boldsymbol{v}_t$ first. We omit the subscript "t" for simplicity, and denote the KG as $\boldsymbol{o}_{\text{KG,full}}$ to distinguish it from the sub-graphs. Since the textual observation, score and knowledge graph are multi-modal inputs, inspired by the VQA techniques [29, 34], we aggregate the inputs by constructing query representation for one modal (or two) to obtain the attention of another modal, through a stacked hierarchical attention mechanism. Fig. 2 shows an overview of our encoder, which consists of two levels. In the high level, we build a query vector from the KG and score, then compute multiple groups of attention values across the components of textual observations. In the low level, we treat the output of the high level as a query, and compute attention values across the sub-graphs.

High-level attention Similar to KG-A2C [3], the KG $\boldsymbol{o}_{\text{KG,full}}$ is processed via GATs [46] followed by a linear layer to get the graph representation $\boldsymbol{v}_{\text{KG, full}} \in \mathbb{R}^{d_{\text{KG}}}$. The score representation $\boldsymbol{v}_{\text{score}} \in \mathbb{R}^{d_{\text{score}}}$ is obtained via binary encoding. While previous works [3, 20] concatenated all observational vectors to form state representation, we build the query vector $\boldsymbol{q}_{\text{high}} \in \mathbb{R}^{d_{\text{high}}}$ by concatenating $\boldsymbol{v}_{\text{KG, full}}$ and

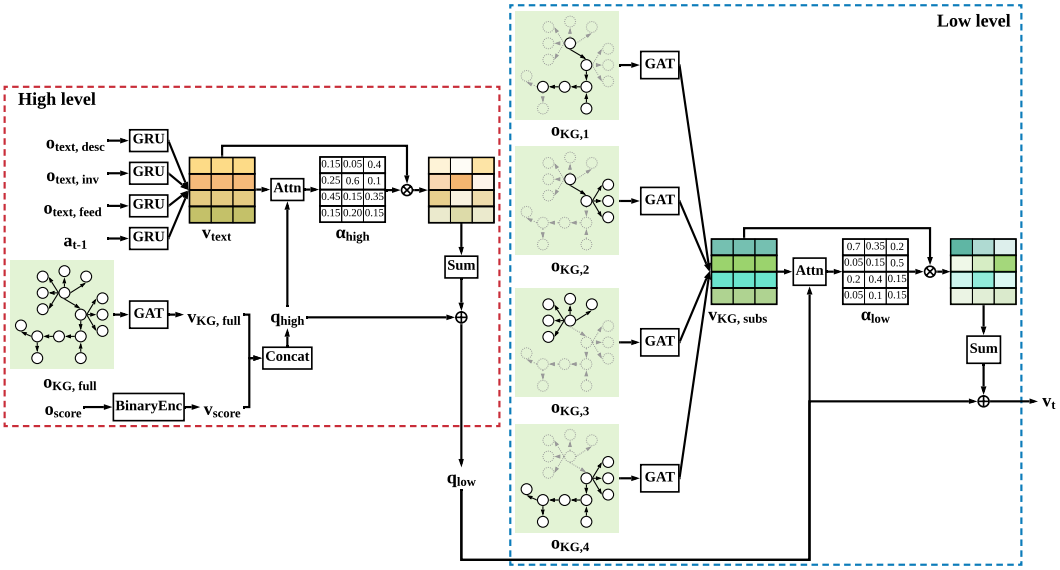

Figure 2: Overview of our stacked hierarchical attention network. In the high level (left), the query vector $\boldsymbol{q}_{\text{high}}$ is the combination of the KG representation $\boldsymbol{v}_{\text{KG, full}}$ and the score representation $\boldsymbol{v}_{\text{score}}$. Then multiple groups of attention values $\boldsymbol{\alpha}_{\text{high}}$ are computed across the components of textual observation $\boldsymbol{v}_{\text{text}}$. In the low level (right), the query vector $\boldsymbol{q}_{\text{low}}$ is the output of high level encoding, and multiple groups of attention values $\boldsymbol{\alpha}_{\text{low}}$ are computed across the sub-graphs. The final output $\boldsymbol{v}_t$ is served as the state representation for action selection.

$\boldsymbol{v}_{\text{score}}$ followed by a linear layer:

$$\boldsymbol{q}_{\text{high}} = \boldsymbol{W}_{\text{Init}}\text{concat}(\boldsymbol{v}_{\text{KG, full}}, \boldsymbol{v}_{\text{score}}) + \boldsymbol{b}_{\text{Init}} \tag{3}$$

where $\boldsymbol{W}_{\text{Init}} \in \mathbb{R}^{d_{\text{high}} \times (d_{\text{KG}} + d_{\text{score}})}$ and $\boldsymbol{b}_{\text{Init}} \in \mathbb{R}^{d_{\text{high}}}$ are weights and biases.

Suppose the textual observation consists of c components[3], we first encode them separately by c GRUs. Instead of concatenating, we consider them individually to build the textual representation vector $\boldsymbol{v}_{\text{text}} \in \mathbb{R}^{d_{\text{high}} \times c}$. Therefore $\boldsymbol{v}_{\text{text}}$ can be treated as multiple image regions or image representation with multiple channels. Inspired by SCA-CNN [11], we compute attention values in channel-wise. However, instead of computing one attention value for each channel, we compute multiple groups of attention values to capture more fine-grained information. Specifically, one group of attention values is computed for each position along the channel:

$$\boldsymbol{\alpha}_{\text{high}} = \text{softmax}(\boldsymbol{W}_{\text{A,high}}\boldsymbol{h}_{\text{high}} + \boldsymbol{b}_{\text{A,high}}) \tag{4}$$

where

$$\boldsymbol{h}_{\text{high}} = \tanh(\boldsymbol{W}_{\text{I,high}}\boldsymbol{v}_{\text{text}} \oplus (\boldsymbol{W}_{\text{Q,high}}\boldsymbol{q}_{\text{high}} + \boldsymbol{b}_{\text{Q,high}})) \tag{5}$$

denotes the intermediate representation and $\oplus$ denotes the addition of a matrix and a vector. $\boldsymbol{W}_{\text{I,high}} \in \mathbb{R}^{d_{\text{high}} \times d_{\text{high}}}$, $\boldsymbol{W}_{\text{Q,high}} \in \mathbb{R}^{d_{\text{high}} \times d_{\text{high}}}$ and $\boldsymbol{W}_{\text{A,high}} \in \mathbb{R}^{d_{\text{high}} \times d_{\text{high}}}$ are weight matrices, $\boldsymbol{b}_{\text{Q,high}} \in \mathbb{R}^{d_{\text{high}}}$ and $\boldsymbol{b}_{\text{A,high}} \in \mathbb{R}^{d_{\text{high}}}$ are biases. This operation is equivalent to dividing $\boldsymbol{v}_{\text{text}}$ as d_{high} sub-vectors $\boldsymbol{v}_{\text{text,sub}} \in \mathbb{R}^{1 \times c}$, then computing channel-wise attention for each of them. The obtained attention values $\boldsymbol{\alpha}_{\text{high}} \in \mathbb{R}^{d_{\text{high}} \times c}$ reflect the multi-positional attentive focus on the textual components. The final step of high-level encoding is to attentively aggregate the query vector with the textual vector. In order to enable multi-level reasoning, we leverage recent advances in attention techniques [17, 28, 53] to learn multi-step reasoning by iteratively updating the query. We first multiply $\boldsymbol{v}_{\text{text}}$ with $\boldsymbol{\alpha}_{\text{high}}$ via dot-product, then sum all the channels and add it to $\boldsymbol{q}_{\text{high}}$ to obtain updated query vector $\boldsymbol{q}_{\text{low}} \in \mathbb{R}^{d_{\text{high}}}$:

$$\boldsymbol{q}_{\text{low}} = \boldsymbol{q}_{\text{high}} + \sum_{i}^{c} \boldsymbol{\alpha}_{\text{high},i} \odot \boldsymbol{v}_{\text{text},i} \tag{6}$$

[3] As discussed in 4.1, c is 4 in this work.

Low-level attention The low-level encoding process is similar to high level, except that the attention values are computed across different sub-graphs. We encode sub-graphs with different GATs and combine them as graph representation $\boldsymbol{v}_{\text{KG}} \in \mathbb{R}^{d_{\text{low}} \times m}$, where d_{low} denotes dimensionality. We treat the output vector of high-level computing as a query vector, and perform linear transformation to ensure $\boldsymbol{q}_{\text{low}} \in \mathbb{R}^{d_{\text{low}}}$. Then we apply the similar attention mechanism of high-level:

$$\boldsymbol{\alpha}_{\text{low}} = \text{softmax}(\boldsymbol{W}_{\text{A,low}} \boldsymbol{h}_{\text{low}} + \boldsymbol{b}_{\text{A,low}}) \tag{7}$$

where

$$\boldsymbol{h}_{\text{low}} = \tanh(\boldsymbol{W}_{\text{I,low}} \boldsymbol{v}_{\text{KG}} \oplus (\boldsymbol{W}_{\text{Q,low}} \boldsymbol{q}_{\text{low}} + \boldsymbol{b}_{\text{Q,low}})) \tag{8}$$

denotes the intermediate representation. $\boldsymbol{W}_{\text{I,low}} \in \mathbb{R}^{d_{\text{low}} \times d_{\text{low}}}$, $\boldsymbol{W}_{\text{Q,low}} \in \mathbb{R}^{d_{\text{low}} \times d_{\text{low}}}$ and $\boldsymbol{W}_{\text{A,low}} \in \mathbb{R}^{d_{\text{low}} \times d_{\text{low}}}$ are weight matrices, $\boldsymbol{b}_{\text{Q,low}} \in \mathbb{R}^{d_{\text{low}}}$ and $\boldsymbol{b}_{\text{A,low}} \in \mathbb{R}^{d_{\text{low}}}$ are biases. Finally, we aggregate $\boldsymbol{q}_{\text{low}}$ with $\boldsymbol{v}_{\text{KG}}$ based on the low-level attention values $\boldsymbol{\alpha}_{\text{low}} \in \mathbb{R}^{d_{\text{low}} \times m}$ to get state representation $\boldsymbol{v}_t \in \mathbb{R}^{d_{\text{low}}}$:

$$\boldsymbol{v}_t = \boldsymbol{q}_{\text{low}} + \sum_{i}^{m} \boldsymbol{\alpha}_{\text{low},i} \odot \boldsymbol{v}_{\text{KG},i} \tag{9}$$

4.4 Action selection and training

Action selection. Given the state representation $\boldsymbol{v}_t$, the action selection can be performed via methods such as template-based scoring [20] and recurrent decoding [3]. In this work, we use the recurrent decoding method to select actions via two GRUs. We first use a template-GRU to predict a template $\boldsymbol{u} \in \boldsymbol{\mathcal{T}}$ based on $\boldsymbol{v}_t$, where $\boldsymbol{\mathcal{T}}$ denotes the template set. Next, we recurrently execute an object-GRU for k steps to decode objects $\{\boldsymbol{p}_i, i \in [1, ..., k]\}$ from the object set $\boldsymbol{\mathcal{P}}$, which is the intersection of the vocabulary set $\boldsymbol{\mathcal{V}}$ and the set of the objects appeared in $\boldsymbol{o}_{\text{KG,full}}$. The probability of an object $\boldsymbol{p}_i$ is conditioned on both $\boldsymbol{v}_t$ and the prediction of the last step (i.e., $\boldsymbol{u}$ or $\boldsymbol{p}_{t-1}$). Finally, the template and objects are combined as action $\boldsymbol{a}_t$.

Training. Our model SHA-KG is trained via the Advantage Actor Critic (A2C) method [37] with a supervised auxiliary task "valid action prediction" [3]. We provide details in the supplementary material.

5 Experiments

We evaluate our method on a set of man-made games in Jericho game suite [20]. We conduct experiments to validate the effectiveness of our sub-graph division and stacked hierarchical attention, and interpret the reasoning and decision making processes.

5.1 Baselines

We use KG-A2C [3] as the building backbone of SHA-KG. Following baselines are considered:

- NAIL [21]: a general agent with hand-crafted rules and pre-trained language models.
- DRRN [20, 22]: a choice-based agent that selects actions from a valid action set.
- TDQN [20]: a parser-based agent with template-based action space.
- KG-A2C [3]: an extension of TDQN with KGs and valid action predictions.

5.2 Experimental setup

Graph construction. The triples for constructing the KG are extracted via Stanford's Open Information Extraction (OpenIE) [6] and two additional rules used in [3]: 1) For interactive objects detected in the current observation, those within the inventory are linked to node "you", while others are linked to the current room. 2) The room connectivity is inferred from navigational actions. Regarding the sub-graph division, we define four graph types: $\boldsymbol{o}_{\text{KG,1}}$ records the connectivity of visited rooms, $\boldsymbol{o}_{\text{KG,2}}$ represents the objects within the current room, $\boldsymbol{o}_{\text{KG,3}}$ represents the objects within the inventory, and $\boldsymbol{o}_{\text{KG,4}}$ is the graph without any connection to "you". $\boldsymbol{o}_{\text{KG,2}}$ and $\boldsymbol{o}_{\text{KG,3}}$ contain the present information only, while $\boldsymbol{o}_{\text{KG,1}}$ and $\boldsymbol{o}_{\text{KG,4}}$ contain both the current and historical information. Besides pre-defined rules, the graph partitioning process can be implemented via automatic methods, which we leave as future work.

Table 1: Raw scores of SHA-KG and baselines. For the baselines, we use the results reported in their original paper [3, 20] except "reverb" and "tryst205" in KG-A2C, which are not reported. $|\mathcal{T}|$ and $|\mathcal{V}|$ denote the size of template set and vocabulary set. **MaxR** denotes the maximum possible score (collected based on walkthrough without step limit). All results are averaged over five independent runs.

Game	$\|\mathcal{T}\|$	$\|\mathcal{V}\|$	NAIL	DRRN	TDQN	KG-A2C	SHA-KG (Ours)	MaxR
acorncourt	151	343	0	**10**	1.6	0.3	1.6	30
balances	156	452	**10**	**10**	4.8	**10.0**	**10.0**	51
detective	197	344	136.9	197.8	169	207.9	**308.0**	360
dragon	177	1049	**0.6**	-3.5	-5.3	0	0.2	25
enchanter	290	722	0	**20.0**	8.6	12.1	**20.0**	400
inhumane	141	409	0.6	0	0.7	3	**5.4**	300
jewel	161	657	1.6	1.6	0	**1.8**	**1.8**	90
library	173	510	0.9	**17**	6.3	14.3	15.8	30
ludicorp	187	503	8.4	13.8	6	**17.8**	**17.8**	150
pentari	155	472	0	27.2	17.4	50.7	**51.3**	70
reverb	183	526	0	8.2	0.3	7.4	**10.6**	50
sorcerer	288	1013	5	20.8	5	5.8	**29.4**	400
spellbrkr	333	844	**40**	37.8	18.7	21.3	**40.0**	600
spirit	169	1112	1	0.8	0.6	1.3	**3.8**	250
temple	175	622	7.3	7.4	**7.9**	7.6	**7.9**	35
tryst205	197	871	2	**9.6**	0	6.7	6.9	350
zenon	149	401	0	0	0	**3.9**	**3.9**	350
zork1	237	697	10.3	32.6	9.9	34	**34.5**	350
zork3	214	564	**1.8**	0.5	0	0.1	0.7	7
ztuu	186	607	0	21.6	4.9	9.2	**25.2**	100

Training implementation. We follow the hyper-parameter setting of KG-A2C [3] except that we reduce the node embedding dimension in GATs from 50 to 25 to reduce GPU cost. We set $\boldsymbol{d}_{\text{high}}$ as 100, and $\boldsymbol{d}_{\text{low}}$ as 50. For both SHA-KG and KG-A2C, the graph mask for action selection is constructed from $\boldsymbol{o}_{\text{KG,full}}$. We denote an action as "valid" if it does not lead to meaningless feedback in a state (e.g., "Nothing happens"). An episode will be terminated after 100 valid steps or game over / victory. For each game, an individual agent is trained for 10^6 interaction steps. The training data is collected from 32 environments in parallel. An optimization step is performed per 8 interaction steps via the Adam optimizer with the learning rate 0.003. All baselines follow their original implementations [3, 20, 21]. To compare with the baselines, we report the average raw score over the last 100 finished episodes during training. We also report the learning curve during ablation study. All the quantitative results are averaged over five independent runs.

5.3 Overall performance

Table 1 shows the performance of SHA-KG and baselines in 20 games. The proposed SHA-KG achieves the new state-of-the-art results in 8 games and is equivalent to the current best baselines in another 7 games. Comparing the three agents using a template-based action space (i.e., SHA-KG, TDQN, KG-A2C), SHA-KG obtains equal or better performance than TDQN and KG-A2C in all of the games, showing the effectiveness of introducing reasoning. The best result of the other 5 games is still achieved by NAIL and DRRN, which apply additional rules and assumptions. NAIL follows nearly fixed rules (e.g., interacting with all the objects, then exploring another room). Though this design principle may be useful for some particular games (e.g., "dragon" and "zork3"), it lacks flexibility so that it performs worse than all learning-based agents in most of the games. DRRN largely reduces the difficulty by selecting actions from the set of admissible actions only. For example, to obtain the first reward "+10" in "acorncourt", the agent has to select a high proportion of complex actions, in which case the assumption of an admissible action set is a large advantage. KG-A2C and our SHA-KG relax this assumption but the action set is as large as $\mathcal{O}(\mathcal{T}\mathcal{P}^2)$, making them infeasible to achieve the first reward stably. However, the reasoning ability still brings improvement compared with the backbone model. While KG-A2C shows worse performance than DRRN in 9 games, SHA-KG outperforms DRRN in 6 of these games, and shows closer scores in other 3 games.

5.4 Ablation study

We perform ablation studies to validate the contributions of different components. We first compare SHA-KG with its three variants with different attention modules:

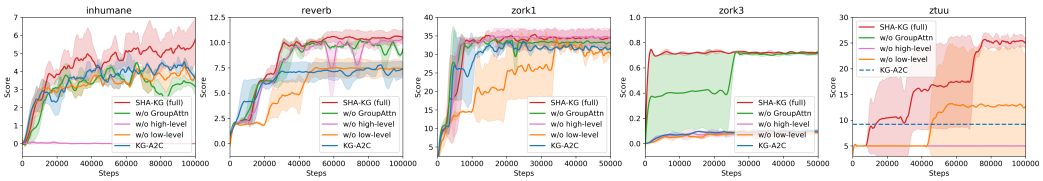

Figure 3: Learning curves of models with different attention modules. The dashed line in "ztuu" denotes KG-A2C's result reported in [3]. The shaded regions indicate standard deviations.

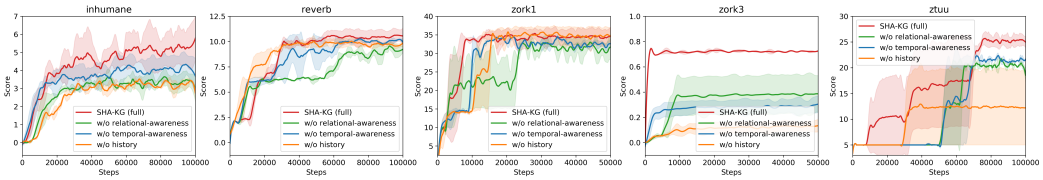

Figure 4: Learning curves of SHA-KG with different types of sub-graphs.

- "w/o GroupAttn": applies single attention value for each channel, i.e., $\boldsymbol{\alpha}_{\text{high}} \in \mathbb{R}^4$, $\boldsymbol{\alpha}_{\text{low}} \in \mathbb{R}^4$.
- "w/o high-level": constructs the initial query vector from $\boldsymbol{v}_{\text{text}}$ and $\boldsymbol{v}_{\text{score}}$, then computes attention values across the sub-graphs. The full KG is not used.
- "w/o low-level": constructs the initial query vector from $\boldsymbol{v}_{\text{KG, full}}$ and $\boldsymbol{v}_{\text{score}}$, then computes attention values across the textual components. The sub-graph division is not used.

Fig. 3 shows the learning curves of 5 games, where following observations can be made: 1) The full model SHA-KG shows similar or better performance than all variants in all cases. Our two-level attention mechanism provides an effective and explainable way to refine information. The first level of hierarchy tells the agent which part of the textual information should be focused on. Based on the output of this level, the second level of hierarchy informs the agent which part of a knowledge graph should be targeting at. 2) The variant "w/o high-level", which considers sub-graphs only but not the full graph (i.e., the pink curve), works for some games (e.g., "reverb" and "zork1") but fails for the others. For the games where the variant is able to achieve the best score, the sample efficiency is also improved, which means that this variant requires fewer interaction steps to achieve the best score. 3) For the variant "w/o low-level" that considers the full KG only but not sub-graphs (i.e., the orange curve), the sample efficiency is a bit low in some games (e.g., "zork1"). 4) The variant "w/o GroupAttn" generally shows a similar performance curve to SHA-KG by considering both the full KG and sub-graphs. However, the performance gap between "w/o GroupAttn" and SHA-KG demonstrates the effectiveness of computing multiple groups of attention values along the channels, which allows SHA-KG to capture more fine-grained information from textual components (in high level) and sub-graphs (in low level). Overall, the learning curves demonstrate that the sub-graph division and stacked hierarchical attention provide complementary contributions to the performance improvement of SHA-KG.

Regarding the contributions of different types of sub-graphs, we further design three variants with different graph partitioning strategies:

- "w/o relational-awareness" combines $\boldsymbol{o}_{\text{KG},2}$ (room objects) and $\boldsymbol{o}_{\text{KG},3}$ (collected objects).
- "w/o temporal-awareness" combines $\boldsymbol{o}_{\text{KG},4}$ with $\boldsymbol{o}_{\text{KG},2}$ and $\boldsymbol{o}_{\text{KG},3}$, respectively.
- "w/o history" removes all historical information.

Fig. 4 shows the results and indicates that the effect of different types of awareness varies with respect to the games. No simple conclusion can be made regarding which type of awareness contributes the most to the final performance (e.g., "w/o relational-awareness" and "w/o temporal-awareness" behave differently in "zork1" and "zork3"). However, considering them collectively and learning to balance their importance lead to the improved performance of our method.

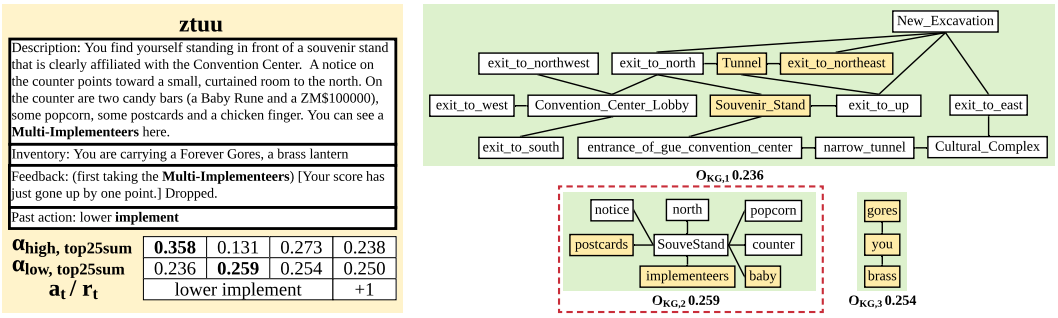

Figure 5: Illustration of the reasoning and decision making processes of the game "ztuu". Left: $\boldsymbol{\alpha}_{\text{high,top25sum}}$ denotes the high-level attention for $\boldsymbol{o}_{\text{text,desc}}$, $\boldsymbol{o}_{\text{text,inv}}$,$\boldsymbol{o}_{\text{text,feed}}$ and $\boldsymbol{a}_{\text{t-1}}$. $\boldsymbol{\alpha}_{\text{low,top25sum}}$ denotes the low-level attention for $\boldsymbol{o}_{\text{KG, 1}}$, $\boldsymbol{o}_{\text{KG, 2}}$, $\boldsymbol{o}_{\text{KG, 3}}$ and $\boldsymbol{o}_{\text{KG, 3}}$. The subscript "top25sum" denotes the sum of 25 largest attention values within a channel (textual observation or sub-graph). $\boldsymbol{a}_t$ is the selected action, and $\boldsymbol{r}_t$ is the reward. Right: the extracted sub-graphs ($\boldsymbol{o}_{\text{KG,4}}$ is omitted due to space limit). The sub-graph with the highest graph-level attention (by SHA) is in the red dashed box. In each sub-graph, the top 3 nodes with the highest attention (by GATs) are highlighted in yellow.

5.5 Interpretability

We interpret the reasoning and decision making process by examining the attentive focus. Although there are controversial points about whether attention values are explainable when being applied to text [25, 51], we believe our attention mechanism is beneficial for interpretability. Compared to word-level attention, the stacked hierarchical attention is conducted in a manner more similar to the region/channel-level attention mechanism, which has been proved to providing interpretability in vision tasks, especially visual-based RL tasks [18, 40] As multiple groups of attention values are computed (e.g., each sub-graph is associated with d_{low} attention values), we aggregate them to facilitate explaining. Specifically, for each channel (e.g., textual component or sub-graph), we sum the top 25 largest values as its attention value, then perform softmax operation across the channels [4]. We denote the obtained attention values as $\boldsymbol{\alpha}_{\text{high, top25sum}}$ for the high level, which correspond to $\langle \boldsymbol{o}_{\text{text, desc}}, \boldsymbol{o}_{\text{text, inv}}, \boldsymbol{o}_{\text{text, feed}}, \boldsymbol{a}_{\text{t-1}} \rangle$, and $\boldsymbol{\alpha}_{\text{low, top25sum}}$ for the low level, which correspond to $\langle \boldsymbol{o}_{\text{KG,1}}, \boldsymbol{o}_{\text{KG,2}}, \boldsymbol{o}_{\text{KG,3}}, \boldsymbol{o}_{\text{KG,4}} \rangle$. Fig. 5 (left) shows a decision making example of the game "ztuu". In the high level (textual components), the agent focuses mostly on the description $\boldsymbol{o}_{\text{text, desc}}$, and then the feedback $\boldsymbol{o}_{\text{text, feed}}$, which is followed by the last action $\boldsymbol{a}_{\text{t-1}}$. All of the three text components contain "implement". In the low level (sub-graphs), the sub-graph of objects in the current room ($\boldsymbol{o}_{\text{KG,2}}$) has the highest attention. Combining both two levels of attention, the agent finally selects the action "lower implement", which receives a positive reward. It shows that the reasoning process enables the agent to select actions leading to positive rewards. Although the GATs in our work are mainly used for obtaining initial graph embeddings instead of assigning attention values, we also visualize the node-level attention within sub-graphs to help understand the reasoning process. Fig. 5 (right) shows three extracted sub-graphs. The digit under the sub-graph denotes graph-level attention. Since the $\boldsymbol{o}_{\text{KG,2}}$ has the highest attention, the agent will focus more on objects it contains. In each sub-graph, nodes with top-3 highest attention (by GATs) are highlighted in yellow. Such node-level attention helps to further constrain (softly) the objects in $\boldsymbol{o}_{\text{KG,2}}$ to derive actions. We conclude that our SHA helps the agent to use information efficiently for taking actions.

6 Conclusion

In this paper, we have studied empowering RL for text-based games with reasoning by exploiting knowledge graphs. We conducted sub-graph division to explicitly introduce relational-awareness and temporal-awareness. Then we designed a stacked hierarchical attention mechanism to obtain effective state representation from multi-modal inputs. Besides obtaining favorable experimental results in a wide range of man-made games, the sub-graph division and attention mechanism enable us to better interpret the reasoning and decision making processes of RL agents.

[4] We also conducted other aggregation methods such as "top10, sum", "top25, mean" and "all, sum", among which we found that "top25, sum" can best interpret the processes. See the supplementary material.

Broader Impact

The high-level goal of this work is to bridge artificial intelligence with human intelligence, cognition and language learning. Researchers in reinforcement learning will benefit from this work by the appropriate use of knowledge graphs. By recording and organizing the information in a structural way, the difficulty of learning can be largely reduced. Besides, KGs can be used to conduct reasoning to interpret the decision making process. Researchers in multi-modal learning will also benefit from the attention mechanism proposed in this work. Although the stacked hierarchical attention is used to aggregate text representation and graph representation, it can also be extended to other forms of inputs such as visual and audio signals. Our work can also be served as an initial study before conducting experiments in real life and with animal / human participants, since it's performed in simulated systems and there's no safety consideration.

Regarding the ethical implications, although currently our work is conducted in games, where language commands are constrained in limited action spaces, for more practical applications this system will be deployed with a richer corpus. From the perspective of language generation, the inappropriate use of generated language commands should be seriously taken into consideration. Another concern lies in the unintended behavior and decision making process, which may lead to dangerous conditions in real world applications. Although we try to improve interpretability through reasoning, there's still a long way to go to make human-AI interaction in a safe and reasonable way.

Acknowledgements

We would like to thank the anonymous reviewers for helpful feedback. This research was partly supported by ARC Discovery Project DP180100966.

References

[1] Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and William L Hamilton. Learning dynamic knowledge graphs to generalize on text-based games. *arXiv preprint arXiv:2002.09127*, 2020.

[2] Leonard Adolphs and Thomas Hofmann. Ledeepchef: Deep reinforcement learning agent for families of text-based games. *arXiv preprint arXiv:1909.01646*, 2019.

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

[4] Prithviraj Ammanabrolu and Mark Riedl. 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, 2019.

[5] Prithviraj Ammanabrolu and Mark O Riedl. Transfer in deep reinforcement learning using knowledge graphs. *arXiv preprint arXiv:1908.06556*, 2019.

[6] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, 2015.

[7] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[8] Timothy Atkinson, Hendrik Baier, Tara Copplestone, Sam Devlin, and Jerry Swan. The text-based adventure ai competition. *IEEE Transactions on Games*, 2019.

[9] Lisa Bauer, Yicheng Wang, and Mohit Bansal. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4220–4230, 2018.

[10] Léon Bottou. From machine learning to machine reasoning. *Machine learning*, 94(2):133–149, 2014.

[11] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6298–6306. IEEE, 2017.

[12] Xiaojun Chen, Shengbin Jia, and Yang Xiang. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, 141:112948, 2020.

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

[14] Daniel C Dennett. The role of language in intelligence. In *What is Intelligence? The Darwin College Lectures*. Cambridge Univ. Press, 1994.

[15] Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. Cognitive graph for multi-hop reading comprehension at scale. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2694–2703, 2019.

[16] Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. What can you do with a rock? affordance extraction via word embeddings. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1039–1045, 2017.

[17] Zhe Gan, Yu Cheng, Ahmed Kholy, Linjie Li, Jingjing Liu, and Jianfeng Gao. Multi-step reasoning via recurrent dual attention for visual dialog. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6463–6474, 2019.

[18] Sam Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. Visualizing and understanding atari agents. *arXiv preprint arXiv:1711.00138*, 2017.

[19] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[20] Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. *arXiv preprint arXiv:1909.05398*, 2019.

[21] Matthew Hausknecht, Ricky Loynd, Greg Yang, Adith Swaminathan, and Jason D Williams. Nail: A general interactive fiction agent. *arXiv preprint arXiv:1902.04259*, 2019.

[22] Ji He, Jianshu Chen, Xiaodong He, Jianfeng Gao, Lihong Li, Li Deng, and Mari Ostendorf. Deep reinforcement learning with a natural language action space. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1621–1630, 2016.

[23] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813, 2017.

[24] Drew Arad Hudson and Christopher D. Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations*, 2018.

[25] Sarthak Jain and Byron C Wallace. Attention is not explanation. 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 3543–3556, 2019.

[26] Vishal Jain, William Fedus, Hugo Larochelle, Doina Precup, and Marc G Bellemare. Algorithmic improvements for deep reinforcement learning applied to interactive fiction. *arXiv preprint arXiv:1911.12511*, 2019.

[27] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. A survey on knowledge graphs: Representation, acquisition and applications. *arXiv preprint arXiv:2002.00388*, 2020.

[28] Zhong Ji, Yanwei Fu, Jichang Guo, Yanwei Pang, Zhongfei Mark Zhang, et al. Stacked semantics-guided attention model for fine-grained zero-shot learning. In *Advances in Neural Information Processing Systems*, pages 5995–6004, 2018.

[29] Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. In *Advances in Neural Information Processing Systems*, pages 1564–1574, 2018.

[30] Wonjae Kim and Yoonho Lee. Learning dynamics of attention: Human prior for interpretable machine reasoning. In *Advances in Neural Information Processing Systems*, pages 6019–6030, 2019.

[31] Bartosz Kostka, Jaroslaw Kwiecieli, Jakub Kowalski, and Pawel Rychlikowski. Text-based adventures of the golovin ai agent. In *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 181–188. IEEE, 2017.

[32] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 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 2822–2832, 2019.

[33] Grace W Lindsay. Attention in psychology, neuroscience, and machine learning. *Frontiers in Computational Neuroscience*, 14:29, 2020.

[34] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances in neural information processing systems*, pages 289–297, 2016.

[35] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 6309–6317. International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[36] David Mascharka, Philip Tran, Ryan Soklaski, and Arjun Majumdar. Transparency by design: Closing the gap between performance and interpretability in visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4942–4950, 2018.

[37] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[38] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013.

[39] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 02 2015.

[40] Alexander Mott, Daniel Zoran, Mike Chrzanowski, Daan Wierstra, and Danilo Jimenez Rezende. Towards interpretable reinforcement learning using attention augmented agents. In *Advances in Neural Information Processing Systems*, pages 12329–12338, 2019.

[41] Keerthiram Murugesan, Mattia Atzeni, Pushkar Shukla, Mrinmaya Sachan, Pavan Kapanipathi, and Kartik Talamadupula. Enhancing text-based reinforcement learning agents with commonsense knowledge. *arXiv preprint arXiv:2005.00811*, 2020.

[42] Karthik Narasimhan, Tejas D Kulkarni, and Regina Barzilay. Language understanding for text-based games using deep reinforcement learning. In *Conference on Empirical Methods in Natural Language Processing, EMNLP 2015*, pages 1–11. Association for Computational Linguistics (ACL), 2015.

[43] Steven Pinker. *The language instinct: How the mind creates language*. Penguin UK, 2003.

[44] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[46] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[47] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Dkn: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*, pages 1835–1844, 2018.

[48] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5329–5336, 2019.

[49] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The World Wide Web Conference*, pages 2022–2032, 2019.

[50] Sam Wenke, Dan Saunders, Mike Qiu, and Jim Fleming. Reasoning and generalization in rl: A tool use perspective. *arXiv preprint arXiv:1907.02050*, 2019.

[51] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. 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 11–20, 2019.

[52] Yikun Xian, Zuohui Fu, S Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 285–294, 2019.

[53] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–29. IEEE, 2016.

[54] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016.

[55] Xusen Yin and Jonathan May. Comprehensible context-driven text game playing. *2019 IEEE Conference on Games (CoG)*, pages 1–8, 2019.

[56] Xingdi (Eric) Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Remi Tachet des Combes, Matthew Hausknecht, and Adam Trischler. Counting to explore and generalize in text-based games. In *European Workshop on Reinforcement Learning (EWRL)*, October 2018.

[57] Tom Zahavy, Matan Haroush, Nadav Merlis, Daniel J Mankowitz, and Shie Mannor. Learn what not to learn: Action elimination with deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3562–3573, 2018.

[58] Mikulas Zelinka, Xingdi Yuan, Marc-Alexandre Côté, Romain Laroche, and Adam Trischler. Building dynamic knowledge graphs from text-based games. *arXiv preprint arXiv:1910.09532*, 2019.

[59] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 793–803, 2019.

[60] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 353–362, 2016.

[61] Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. Variational reasoning for question answering with knowledge graph. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[62] Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. Knowledge graph embedding with hierarchical relation structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3198–3207, 2018.