Natural Language Understanding for Learning Agents

by

Sukai Huang

ORCID: 0000-0001-7886-5571

A confirmation report for the
degree of Doctor of Philosophy

in the
Faculty of Engineering and Information Technology
School of Computing and Information Systems
THE UNIVERSITY OF MELBOURNE

August 2022

Contents

Chapter 1

Introduction

Natural language enables humans to encode abstract ideas, generalise concepts and convey intentions to others [Gopnik and Meltzoff, 1987]. An artificial learning agent, which trains its policies via interactions with its task environment, would likewise benefits from these capabilities – encoding and communicating prior knowledge and experiences. However, agents trained using classic learning paradigms such as Reinforcement Learning (RL) appear to lack such capabilities, making it difficult to learn efficiently and generalise their knowledge to unseen tasks. This is the case despite the fact that in recent years, RL agents combined with deep learning techniques have already surpassed human experts in complex video games [Badia et al., 2020, Berner et al., 2019]. Specifically, previous studies have revealed two typical issues with Deep Reinforcement Learning (DRL) agents: the first is sample inefficiency – In contrast to humans, who are able to learn in unfamiliar environment because of their rich prior knowledge about the world [Spelke and Kinzler, 2007], RL agents would require 100,000 times more data to learn the same thing. Even worse, RL agents may fail to learn if the environment offers sparse reward signals [Hafner et al., 2020]. The second issue is poor generalisation ability – modern learning agents typically struggle to extend their policies to new environments, even if the objectives and work contents are identical. For example, previous studies have shown that RL agents often requires a whole set of re-training when the task environment is slightly altered [Berner et al., 2019].

With recent breakthroughs in representation learning for language, we now have models that can acquire general knowledge by pre-training large text corpora and then utilise that prior knowledge in unfamiliar downstream decision making tasks. [Devlin et al., 2018, Brown et al., 2020] Thus, it brings light to the potential of utilising large-scale pre-trained natural language understanding (NLU) models for learning agents in order to improve sample efficiency and generalisation performance. Most importantly, many textual corpora online, whether general or task specific, are underutilised. They are valuable for making decisions, but methods for learning agents to utilise these textual information has not been explored thoroughly in previous works.

There have been several attempts at incorporating language data into decision-making tasks with learning-based agents (e.g. training a game-playing agent) to facilitate learning. Text corpora used in previous studies can be categorised into task-specific corpora (e.g., game walkthroughs, manuals) and general corpora (e.g., Wikipedia corpus, BookCorpus). A common application of task-specific corpora is reward shaping – using text data to engineer an auxiliary reward function to provide more detailed feedback on agent actions [Goyal et al., 2019, 2020]. As for general text corpora, previous work has also examined the transferability of a generic pre-trained language model fine-tuned into Offline RL model and reported consistent improvements on efficiency and performance [Reid et al., 2022]. To best of our knowledge, only few attempts at language-assisted RL approach have been made and it is an open question about how to make it effective in enhancing efficiency and generalisation. Improving language-assisted RL is beneficial for solving not only computer games, but also other decision-making tasks that want to use learning-based agents (e.g., Generative Chatbots, AI assistants). In addition, this research may be even appealing to language-assisted human learning or the related fields in cognitive science. It is because "instruction following" – a more common phrase – is also one of the common challenges that a toddler faces for their cognitive development [Carey and Bartlett, 1978, Sharp and Delmonte, 2015]. Therefore, we hope that this research could in the end produce some valuable insights for the study of toddler cognitive development.

In particular, language-assisted RL approach raises two key challenges that are under explored:

- Parsing different abstraction levels of natural language
- Multimodal alignment and fusion

While existing work has shown success in making agents learn faster using task-specific language data, an important research gap is that texts were treated as having a unified level of abstraction. However, language information often varies along the abstract-concrete dimension and information at different levels has different influences on decision-making. Building a learning-based agent that is capable of mapping text data based on granularity and computing corresponding rewards with appropriate forms is essential for efficiently utilizing text information.

Learning-based agents typically only deal with one modality of data. With the presence of text data, the task falls into the category of multimodal machine learning. However, it is still under explored how learning-based agents can effectively align and fuse language sentences, visual observations and sequences of past actions for decision-making. In particular, given a sequence of text guidelines, the language assisted RL agent needs to know how to segment text information into a useful sequence of subgoals and then know what actions need to be executed to fulfill the subgoal and also detect each sub-goal is completed.

1.1 Research Questions

To this end, the following research questions are investigated:

1. **How to incorporate different multi-modalities of data under the language-assisted reward shaping method?**

   ***RQ**$_{1a}$: How to align task-specific natural language sentences with segments of agent trajectories?*

   Reward shaping, which involves turning information into immediate auxiliary rewards, was commonly used in previous work on language assisted RL approach. In this setting, a reward shaping module is created to judge whether the agent is listening to the provided language instructions, and then delivers additional immediate rewards accordingly. Specifically, a typical reward shaping module will try to align natural language sentences with the agent trajectories so that it can a). predict whether the agent behaviour follows the instruction and b). track the agent's progress towards subgoals stated in the instruction sentences. The training of this module can be done separately from the training of the agent's policy network and is done in a supervised learning setting in which a collection of instructions and demonstrations pairs are prepared beforehand. However, there are several limits on the capacity of the existing approach. Our first concern is that the language reward module is stateless, which limits the generality and can result in inaccurate judgements on the distribution of rewards [Goyal et al., 2019].

   Including world states in the process of rewards shaping is essential for delivering more accurate rewards. However, introducing another modality of data (visual) would make alignments between natural language sentences and the agent trajectories considerably more difficult. The alignment problem in multimodal machine learning is still an emerging research topic and there is no widely accepted standard. The challenge lies in the fact it is very expensive/labor intensive to obtain a multimodal dataset with explicit alignment annotation. Therefore, several works have been conducted to learn alignment implicitly from unlabeled multimodal dataset. In prior work on visual-language models, contrastive learning has been used to implicitly learn the alignment in an unsupervised way [Radford et al., 2021, Jia et al., 2021]. This includes generating negative samples from different modalities of data that are not paired with each other. It has been shown that sequence models learnt via a contrastive loss possess implicit alignment knowledge among different modalities of data. Other than contrastive learning, Zhang et al. [2021] trained a Transformer-based model [Vaswani et al., 2017] to predict if the video is related to a query or not. By having this training process, it was found that the temporal alignment information between the query vectors and relevant visual features can be obtained from the Transformer's attention coefficient.

Nevertheless, extending the existing implicit multimodal alignment methods to the task of agent learning in interactive environments is challenging as it lacks large-scale off-the-shelf instructions and demonstration datasets while generic visual-language models learn their alignment capability from unprecedented large image-text pairs (e.g., 1.8B in Jia et al. [2021]). In addition to that, alignment task itself becomes more difficult because of two factors: 1) there is another modality added in, which makes fusion more challenging and 2) it forces the alignment to reason about the whole trajectory and thus force the model to produce more accurate prediction. As a result, further research is required in order to effectively align agent trajectories (visual + actions) with natural language instructions.

***RQ*$_{1b}$*: What is the best way of handling multimodal data for learning-based agents?*

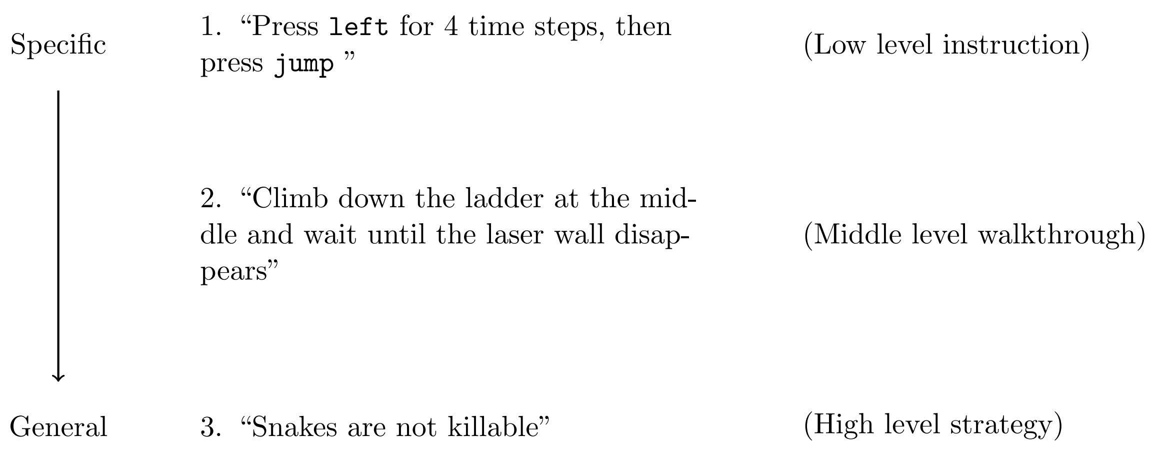
Instead of using language to provide feedback for training, we would like to explore using language to extend the representation of the state space such that more succinct policies can be learnt. In order to accomplish this, the policy network must fuse multiple unimodal representations (visual observations and language data) into one joint multimodal representation. To the best of our knowledge, Attention mechanism [Vaswani et al., 2017] would be the most common multimodal fusion method for most of the visual-language models [Li et al., 2019, Lu et al., 2019, Alayrac et al., 2022]. However, it is not guaranteed that Attention will also suit under RL paradigm because RL paradigm differs significantly from unsupervised learning paradigm in which those generic visual-language models get trained. Due to the exploitation-exploration trade-off, agents are not guaranteed to gather a diverse range of training dataset. Also the i.i.d. assumption of training datasets is no more applicable. Thus, the performance of Attention-based fusion method may be affected. Moreover, Attention-based fusion methods may exacerbate sample inefficiency of RL models as they are data-hungry [Hessel and Lee, 2020], and

Pham et al. [2020] also pointed out that Attention-based model can be insensitive to the temporal logic appeared in the language data. Therefore, it is worth trying different fusion methods such as Low Rank Fusion [Liu et al., 2018] to find out which one fits better.

2. **How to effectively deal with abstraction levels of natural language instructions under the language assisted reinforcement learning paradigm?**

   RQ*$_{2a}$*: Taking into account the levels of abstraction, how can language information be concretised into steps of actions?

   In ***RQ*$_{1b}$** we would take a flat representation, and ***RQ*$_{2a}$** will study the hierarchical one. In natural language, details and attributes are often left out in order to communicate more general and task-independent information. Figure 1.1 shows examples of game walkthroughs at different abstraction levels from the famous platform adventure game Montezuma's Revenge, in which the objective of this game is to exit the level by searching for adequate keys and jewels and killing enemies along the way.



Specific	1. "Press `left` for 4 time steps, then press `jump` "	(Low level instruction)
↓	2. "Climb down the ladder at the middle and wait until the laser wall disappears"	(Middle level walkthrough)
General	3. "Snakes are not killable"	(High level strategy)

FIGURE 1.1: Examples of natural language instructions for Montezuma's Revenge, the first instruction only pertains to one particular game situation and is task specific. Moving on to the second and third walkthrough sentences, the information becomes more general and transferable across various game levels.

Information at different abstraction levels has varieties of impacts on agents. When it comes to low-level instructions, a common way to utilise it is to directly align with concrete action sequences. However, in a task environment that changes often, low level instructions may be ineffective if blindly followed, as they may prevent agents from adapting to the changes. Furthermore, the difficulty of comprehending a low-level instruction will increase if the length of the instruction gets longer. Several factors contribute to this:

(a) The number of subgoals is likely to increase as the instruction gets longer, thereby making the task more complex;

(b) Events may not be described in a monotonic order if the instruction gets longer, thus aligning agent trajectories to instruction becomes more difficult;

(c) There may be chit-chat obscuring key details [Kiseleva et al., 2022].

On the other hand, high level strategies capture the big picture of the problem, making them robust to variances. But it is not trivial to convert abstract strategies into concrete agent actions, and agents must possess solid prior knowledge in order to comprehend them. In Goyal et al. [2019]'s work, one of our concern was that the authors examined their method with super simple language, specific only to 3 seconds snippets of game-play. This isn't going to exhibit much in the way of multi-stage instructions, strategy, and other interesting quirks of language. So it remained unclear whether the current language reward shaping method could handle complex language data at different abstraction levels. In order to answer this question we will carry a comprehensive study of the how best different abstraction levels can be used to learn reward functions and extend state representations.

$\boldsymbol{RQ}_{2b}$: *How to store past experiences/knowledge in the form of natural language and transfer to new decision-making tasks?*

RL transfer learning is still an emerging topic due to the diversity among different environments. A major challenge comes from the fact that knowledge may take a variety of forms, which must be transferred differently [Zhu et al., 2020]. With the rise of language-assisted RL approach, we see the potential of saving knowledge in text form, which can then be transferred into new tasks, assuming that language-assisted RL agent is capable of parsing the text information. The use of natural language as the medium to encode knowledge can effectively address the issue of high variances among different task environments. It is because natural language serves as an abstract layer to the transferable knowledge, making it possible for trajectories, model dynamics, policies and even the value functions to be shared regardless of the differences in internal implementations and formats. In order to answer this question, we will start with a study on how to encode knowledge from agent policy networks as the form of natural language and exploit it for new tasks and also conduct a comprehensive study on how language improves generalisation of learning based agents.

1.2 Research Scope

Data & Learning Environment Due to data availability, the task environment of our research will be video games for which English game walkthroughs can be sourced from public websites. We will stick to games that have a discrete action space. We plan to start our research on Atari Montezuma's Revenge game (See Figure 1.2), the well-known adventure platform game that many modern RL agents failed to solve. At a later stage, we may extend our research to a harder game – Nethack [Hambro et al., 2022] (See Figure 1.3). In terms of the language dataset, the instruction and demonstration data collection for Montezuma's Revenge from Goyal et al. [2019] will be used. We will try to extend the Montezuma's Revenge dataset with longer and more complex text data such as walkthroughs from online websites. For Nethack environment, the language data will mostly come from the official guidebook as well as the in-game messages. The guidebook contains a lot of general information such as "how to fight monsters" or "item attributes", which serves as a very good test bed for methods of comprehending abstract information.

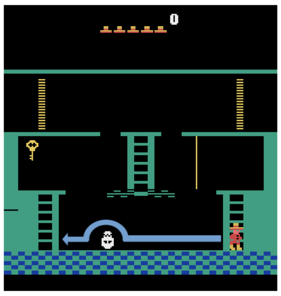
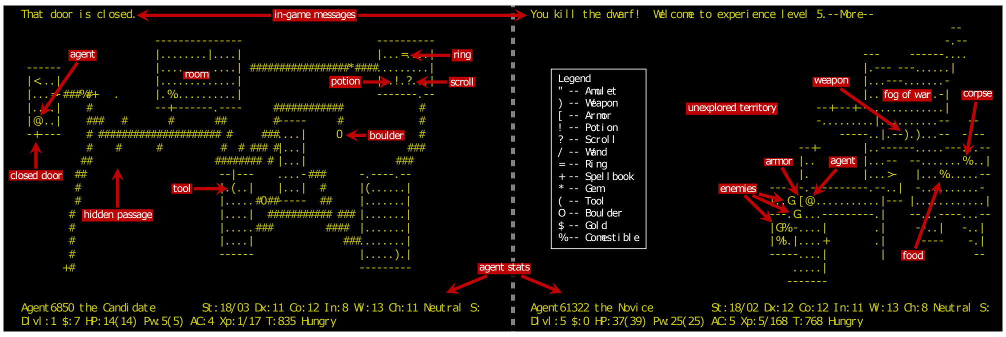


Figure 1.2: Credit by Goyal et al. [2019], Atari Montezuma's Revenge Environment



```
That door is closed.
You kill the dwarf!  Welcome to experience level 5.--More--

Legend
" -- Amulet
) -- Weapon
[ -- Armor
! -- Potion
? -- Scroll
/ -- Wand
= -- Ring
+ -- Spellbook
* -- Gem
( -- Tool
0 -- Boulder
$ -- Gold
%-- Comestible

Agent6850 the Candidate        St:18/03 Dx:11 Co:12 In:8 Wi:13 Ch:11 Neutral  S:
Dlvl:1 $:7 HP:14(14) Pw:5(5) AC:4 Xp:1/17 T:835 Hungry

Agent61322 the Novice          St:18/02 Dx:12 Co:12 In:11 Wi:13 Ch:8 Neutral  S:
Dlvl:5 $:0 HP:37(39) Pw:25(25) AC:5 Xp:5/168 T:768 Hungry
```

Figure 1.3: Credit by Küttler et al. [2020], NetHack Learning Environment

Models Our model design will be inspired by latest Visual Language models (e.g., Flamingo [Alayrac et al., 2022]), Visual Question Answering (VQA) models (e.g., RETRO [Borgeaud et al., 2022]) as well as Hierarchical RL techniques (e.g., Option MDP [Sutton et al., 1999, Andreas et al., 2017]).

Methods We will test our idea about language-assisted methods for learning-based agents mostly in reward shaping RL paradigm [Ng et al., 1999]. The agent will be trained using IMPALA [Espeholt et al., 2018], an Actor-Critic Off-Policy Policy Gradient algorithm that was deemed to have better exploration strategy [Weng, 2018]. Due to lack of exact alignment labels between language instructions and agent demonstrations, we will only focus on weakly supervised learning methods (e.g., contrastive learning using positive and negative sample pairs) that implicitly learn multimodal alignments.

Evaluation Confusion matrix-based metrics will be used for evaluating language reward shaping module. In terms of the performance of language-assisted learning-based agents, we will count the success rate of episodes to evaluate the effectiveness of the proposed methods. And we will also count the difference between the number of steps to reach a goal state with or without the language-assisted RL method so as to evaluate to what extent the proposed method improves the sample efficiency.

Chapter 2

Literature Review

2.1 Language-assisted RL

Previous studies [Berner et al., 2019, Hafner et al., 2020] has shown that modern deep reinforcement learning models suffer from sample inefficiency and low generalisation issues – a modern RL model may need hundreds of thousands of episodes to learn and be extremely unstable to environment changes. Therefore, several studies have been conducted on language-assisted RL approach that aims to improve sample efficiency and generalisation performance by utilising supplementary language information, either task-specific (e.g., instructions, manuals) or general, to facilitate learning.

2.1.1 Reward shaping by using instructions

Reward shaping allows the RL agent to avoid from the sparse reward problem and thus to learn a good (optimal) policy faster. For example, let us consider a learning environment where an agent gets positive reward signals only when it reaches the goal. Under this circumstances, an agent has to conduct a whole new avenue of clueless exploration so as to find the right path, which is very time-consuming and sample inefficient. Reward shaping, however, involves constructing an additional reward function so that agents can receive more informative and frequent reward signals during training, thereby smoothing the learning curve. Specifically, reward shaping applies to problems that can be expressed as a Markov Decision Process (MDP) tuple $M = \langle S, s_0, A, T, R, \gamma \rangle$ where:

- S is a (finite) set of **states**;
- $s_0 \in S$ is the **initial state**;

- $A = \{a_1, ..., a_k\}$ is a set of $k \geq 2$ **actions**;
- $T(s'|s, a)$ represents the **transition probabilities** of transiting to next state s' upon taking action a at current state s;
- $R(s, a) \in (0, \infty)$ is the immediate **reward** returned by applying action $a \in A$ at state s;
- $\gamma \in [0, 1)$ is the **discount factor** which controls the importance of rewards over time. When $\gamma = 0$, the agent concerns about the immediate reward only; when $\gamma = 1$, the agent concerns about all the future rewards equally.

Reward shaping approach runs the RL algorithm on a transformed MDP $M' = \langle S, s_0, A, T, R', \gamma\rangle$, where $R' = fusion(R, F)$ is the reward function in the transformed MDP, and $F : S \times A \times S \to \mathbb{R}$ is called the shaping reward function.

Language Reward shaping RL is one approach as described by Luketina et al. [2019]. In particular, task-specific natural language information will be converted to immediate extra reward signals to guide the agent at each single state. From a macro perspective, language information implicitly sets up several goal states for the problem, and thus turns a vanilla MDP into Stochastic Shortest Path Problems (SSP), a MDP variant that generalises the classic deterministic shortest path problem [Bertsekas, 2012] and can be expressed as the tuple $\langle S, s_0, A, T, R, \gamma, G\rangle$, where S, s_0, A, T, R and γ are defined the same in the MDP setting while $G \subset S$ is a set of goal states. The essential difference is that the policy of SSP aims to maximise the expected long term reward for MDPs and minimise the expected cost to reach a goal state for SSPs. From a micro perspective, the task-specific language information approximates subgoal-based heuristic to craft shaping reward function F, which will guarantee optimality as proved by Ng et al. [1999].

Goyal et al. [2019] proposed a concrete Language Reward Shaping RL framework that uses low-level instructions to construct immediate rewards for game playing agents. The method has shown improvements on the success rate of solving simple tasks compared to traditional RL. To be more specific, they collected a Montezuma's Revenge demonstration dataset containing trajectory clip (3 seconds each) annotated with short and simple language descriptions ($\leq$ 20 words each). After that, they trained an extra reward shaping network aside from the RL policy network to predict whether a given trajectory aligned with the instructions. The alignment score is between 0 to 1 (0: not aligned at all $\to$ 1: perfectly aligned). In the agent training phase, the shaping reward function F is constructed in the following way:

1. A complete instruction l that covers the whole (small) task is provided;

2. (a_t) denotes the action sequence $(a_0, a_1, ..., a_t)$ starting from time step 0 until t, $A_{score}((a_t), l)$ denotes the alignment score by feeding action sequence a_t and language instruction l into the pre-trained reward shaping network;

3. The shaping reward function is $F = A_{score}((a_t), l) - A_{score}((a_{t-1}), l)$

4. The final immediate reward function is $R' = R + F$ (additive fusion)

However, in Goyal et al. [2019]'s experiment, although the reward shaping module is trained on the three-second Montezuma's Revenge demonstration clips, he did not further evaluate his RL agent by letting it solve the whole Montezuma's Revenge game. Instead, the reward shaping RL agent was evaluated by asking it to perform some simple tasks in a hand-picked room, where the task can be described in one sentence, such as "go to the left and then go down the ladder". Therefore, during the evaluation phase, the instruction l is exactly the task description. The final immediate reward function become $R' = 0 + F$ because the rewards from the original game environment are no more relevant.

As such ,there are some aspects that are not covered by Goyal et al. [2019] – firstly, their approach is only examined on short and simple low-level instructions. This isn't going to exhibit much in the way of multi-stage instructions, strategy, and other interesting quirks of language. Secondly, the reward shaping network of their model relied only on actions but neglected states information, which could lead to imprecise shaping rewards.

2.1.2 Early Fusion of text information

Hu et al. [2019] proposed a language-assisted RL model that directly treated text information as observation data from the task environment. Separate observation data modalities (e.g., language, visual) were integrated in this case into a unified multimodal latent representation before proceeding to the policy network (i.e., "early fusion"). This type of handling text data – treating texts as a part of observations – is particularly widespread in the study of Grounded Language Learning. It is an emerging research area that aims to train natural language understanding (NLU) of agents/robots by interacting with an environment. It was argued that humans learn NLU in the exact way in their early childhoods [Hill et al., 2020]. Although this area is not a focus of this thesis, we can certainly get inspired from their work because both language-assisted agent learning and Grounded Language Learning can be regarded as almost the same problem – "instruction following in an interactive environment". A common principle among previous studies on Grounded Language Learning [Hill et al., 2020, Cao et al., 2020, Kiseleva et al., 2021, Hermann et al., 2017] is that they all attempted to implement complex sequence models (e.g., LSTM [Hochreiter and Schmidhuber, 1997] variants, Transformer variants) to build a memory module for learning agents. Its role in this setting is to construct a latent variable that integrates agent's observations over time. This may bring light to the potential of enhancing language-assisted RL approach by adding a memory mechanism. In fact, some preliminary research has been conducted on RL agents that incorporate episodic memory to solve tasks [Ramani, 2019]. But there are also some arguments about the effectiveness of this approach. For instance, Hu et al. [2021] argued that modelling a critic network that estimates the Q value can be treated exactly as modelling a episodic memory table.

2.2 Decision-making guided by general language information

Language information can be classified along the abstract-concrete axis (i.e., Levels of Abstraction) [Blank et al., 1978, Rasmussen, 1986]. This is due to the fact that humans often induce, which is to derive a general principle from specific observations. In this way, we communicate ideas and concepts that are abstracted away from the reality. Details and attributes under a certain level of abstraction will be removed so as to communicate more general and task-independent information.

This concept is crucial in language-assisted RL because the circumstances of handling language information at various abstraction levels can be much different (See $\boldsymbol{RQ}_{2a}$ in Section 1.1) and thus it would be worth trying to handle them separately. Moreover, the capability of comprehending high-level information and concretising into actions is critical to improving generalisation performance of RL agents because general knowledge that can be shared across different tasks are often encoded by abstract language.

Nevertheless, I am going to mention some of previous studies that aimed at utilising general text information to guide decision-making process.

2.2.1 Option MDP

Option MDP model was first introduced by Sutton et al. [1999], which solves a problem by maintaining a set of sub-policies $\{\pi_0, ..., \pi_i\}$. A policy π_i in Option MDP model is a mapping from states S to actions plus a stop signal $A \cup \{\texttt{STOP}\}$. At the starting point, a sub-policy network π_m will be activated and then return actions for the problem until the `STOP` symbol is emitted, at which point control is passed to the next sub-policy $\pi_n, n \neq m, n \in \{0, .., i\}$. Researchers have tried to use general strategies information in natural language format to formulate Option MDP model (a “general strategy” in video game context means some useful guidance on the game-play level, where the details about how to exactly win a specific game level is not mentioned). Specifically, general strategy information was used to a). decide the number of sub-policies of the agent and to b). derive a rule on how to select the successive sub-policy. For example, a typical “general game stratey” may contain information about what skills are needed in this game. Assuming that this game requires map navigation skill, combat skill and puzzle-solving skill, then based on that, three sub-policies are created and assigned to the agent. Each dedicated sub-policy is activated once the agent encountered that specific problem. Results from Andreas et al. [2017], Jiang et al. [2019] show that options MDP model in this manner is able to achieve better zero-shot generalisation performance than singular-policy RL models. It makes sense because with the use of Option MDP model, an unseen task can be decomposed into separated familiar sub-tasks and thus can be addressed by reusable sub-policies. This approach is not limited to “general natural language data” as long as the language data is able to give clues about how to construct sub-policies. However, applying Option MDP on low-level instructions might be prone to overfitting. For instance, given an instruction “go left and then climb up”, it may not be wise to assign two sub-policies to control “move left” action and “move up” action respectively. Having an overfitted modular structure make it hard for agents to make decision on dispatching.

2.3 Multimodal Alignment

Multimodal Alignment is the task of identifying and modeling cross-modal connections between all elements of multiple modalities, where multimodal data is presented in diverse qualities, structures and representations. This is the sub-task under the language reward shaping RL approach, where the reward shaping module is required to align natural language sentences with the agent trajectories so that it can predict whether the agent behaviour follows the instruction and track the agent’s progress towards subgoals stated in the instruction sentences. Nevertheless, alignment task is prevalent in other multimodal machine learning research areas.

2.3.1 Text Video Alignment

Text Video Alignment aims at establish temporal correspondence between subtitles and continuous sequence of video frames. In the modern text video alignment models, Transformer is the most commonly used architecture for conducting cross attention mechanism over text and video data [Zhang et al., 2021, Bull et al., 2021, Zhao et al., 2021]. Based on that, some tried to either over-complicate the model to enhance temporal modelling [Li et al., 2021] or simplified the model (e.g., use LSTM or even KNN instead of Attention) to promote data-efficient alignment [Wang et al., 2021, Fan et al., 2021].

While some studies directly modeled the problem as binary frame-wise prediction of whether the current frame corresponds to the query text [Bull et al., 2021], most other work focused on conducting weakly supervised learning so as to alleviate the demand for expensive and labour-intensive frame-level data annotation [Zhang et al., 2021, Li et al., 2021, Song et al., 2016]. For example, Zhang et al. [2021] trained a Visual Question Answering (VQA) model for a multi-label classification problem and found that the temporal alignment information can still be obtained from attention coefficients of the Transformer architecture.

In addition, Song et al. [2016] proposed a novel method that focused on extracting actions from the video frames to match with the verbs in language data, and it showed better alignment under the weakly supervised learning paradigm. The way of crossing connecting language, visual and actions in this study looks closely relevant to our research topic.

2.3.2 Connectionist Temporal Classification (CTC) in Automatic Speech Recognition (ASR)

The lack of aligned dataset is also a well-known issue in the field of Automatic Speech Recognition (ASR). In order to alleviate the demand for detailed data alignment annotation, Connectionist Temporal Classification (CTC) is designed for aligning each character to its location in an audio sequence with the need of aligned training dataset. CTC trains the alignment model by increasing the probability of all the possible alignment patterns that lead to the ground truth output [Graves et al., 2006].

It turns out that CTC is a generic method for multimodal alignment problem, as long as the the alignment of input to target sequences satisfies the following properties:

1. The alignment of input to target sequences are monotonic;
2. The alignment of input to target is assumed to be “many-to-one”;
3. The third property can be inferred from the preceding two: the length of input sequences must be greater than target length.

We expect to see the potential of applying CTC method in our research topic, provided that we are able to carefully model our task of aligning from agent trajectories (input) to natural language instructions (output) to comply with the three CTC properties.

2.3.3 Visual Language Model

Visual Language Models that are based on Transformer architecture were introduced recently [Li et al., 2019, Lu et al., 2019]. Transformer architecture provides a simple way to handle multimodalies of data because the Attention block inside it allows to take q, k, v three vectors to do complex fusion. In the most common case, q is assigned with text data while $k = v =$ image. Visual Language Models that take image sequences and text sentences as inputs are able to learn the alignment implicitly (demonstrated by Zhang et al. [2021]). Therefore, investigating how to improve the performance of Visual Language Models may also be helpful in solving multimodal alignment task.

Current studies focus on two aspects of improving the model:

1. Carefully designing the architecture to embed pre-trained visual and language encoders [Alayrac et al., 2022];

2. Contrastive learning with large and noisy image and alt-text data pairs [Radford et al., 2021, Jia et al., 2021].

In fact, knowledge transfer from pre-trained models and data augmentation technique could help not only the multimodal alignment but also the decision-making process. More details can be found in Section 2.5.

Although Transformer was used as the fundamental component of most modern Visual Language Models, it encodes the order information in a very roundabout way (positional encoding [Su et al., 2021]). This is thus a commonly suspected reason why Transformer might be insensitive to the temporal logic appeared in the language data. Pham et al. [2020] has shown that the vanilla Transformer model trained by GLUE benchmark [Wang et al., 2018] is not sensitive to word ordering (See Figure 2.1). Therefore, Transformer may not be the best model for aligning agent trajectories to language instructions because it may fail to capture the temporal logic of the events.

2.4 "Instruction Understanding and Following" in other Domains

Given the fact that we focuses on the task of utilising language data to facilitate learning of game-playing agents, the studies on the task of "instruction understanding and following" in other domains may also be able to bring some novel insight to our research.

2.4.1 Dialogue System

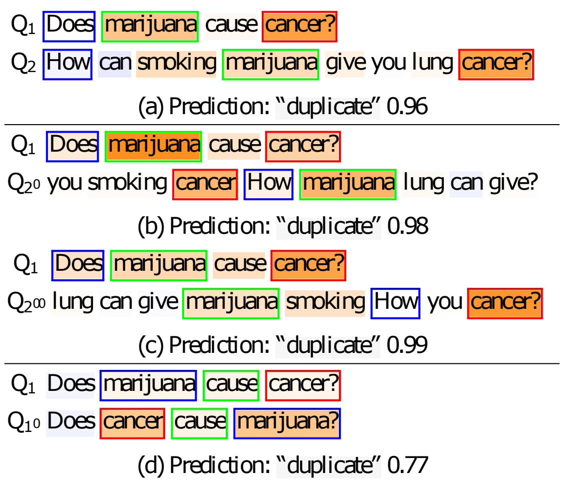
The language assisted game-playing AI and the chatbots may share the similar characteristics because both the intelligent agents need to do a sequence of decision-making with the use of language data.

Similar to Grounded Language Learning models, modern deep learning dialogue systems often focus on a) building a memory module to encode past experiences and also b) explicitly modelling knowledge of the world as a graph network (neuro-symbolic model will be discussed in Section 2.4.2) [Bara et al., 2021, Andreas et al., 2020, Yang et al., 2022].

In fact, most of the dialogue system studies stick with the conventional semantic parsing method – a typical pipeline is 1) semantic parsing of the user's conversation into logical forms; 2) deriving database queries from logical forms and finally 3). retrieving responses with pre-defined patterns. More details about semantic parsing for planning will be discussed in Section 2.4.3.

2.4.2 Visual Reasoning Models, Neuro-symbolic AI and Model-based RL

The visual reasoning studies is highly relevant to our topic because firstly, visual reasoning models often take video segments and questions in natural language as input and answer the question by choosing a option from a list of possible answers (i.e., MCQ), so



Q₁ Does marijuana cause cancer?
Q₂ How can smoking marijuana give you lung cancer?
(a) Prediction: "duplicate" 0.96
Q₁ Does marijuana cause cancer?
Q₂₀ you smoking cancer How marijuana lung can give?
(b) Prediction: "duplicate" 0.98
Q₁ Does marijuana cause cancer?
Q₂₀₀ lung can give marijuana smoking How you cancer?
(c) Prediction: "duplicate" 0.99
Q₁ Does marijuana cause cancer?
Q₁₀ Does cancer cause marijuana?
(d) Prediction: "duplicate" 0.77

FIGURE 2.1: Credit by Pham et al. [2020], in a classification task that predicts whether two questions are duplicate, a RoBERTa-based model could achieve a high 91.12% accuracy (shown in a). The authors observed that the prediction remains the same when they randomly shuffle the words (shown in b-c). The model also tended to stick to the original answer even when the shuffling change the semantic meaning (shown in d).

it falls under the category of multimodal learning; secondly, game-playing agents also need to reason over observations (which are often in visual format) so as to come up with good decisions.

When it comes to building visual reasoning models, there are two main camps. In one, all-in-one Transformer model is good enough to implicitly extract association or even causality knowledge and meanwhile it avoids the labour intensive work of handcrafted feature engineering. Ding et al. [2021] proposed a transformer-based model and achieved higher accuracy (75.6% vs 46.5%) and less training data (∼40%) compared to baseline model in the CLEVRER visual reasoning benchmark [Yi et al., 2019]. They claimed that their model relied on three key aspects:

1. Soft-discretisation: given that objects are atomic units of Newtonian physics interactions, the author used pre-trained object detection model to discretise the pixel format images to the level of objects;
2. Self-attention across each discrete entities so as to implicitly extract association and causal relationship;
3. Masked auto-encoding (MAE [He et al., 2022]) self-supervised learning approach: they mask random entities of the input and train the model to reconstruct the missing entities (See Figure 2.2).

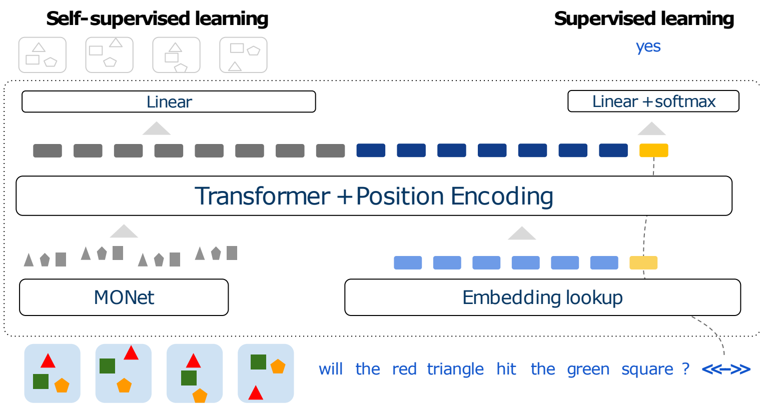


FIGURE 2.2: Credit by Ding et al. [2021], a schematic of the transformer-based visual reasoning model architecture

In the other camp, researchers proposed Neuro-Symbolic model for which a typical architecture would consist of a graph network to model knowledge of the world and a deep learning based perception system. Neuro-Symbolic AI explicitly defines some rules or structures (no need to be strong) through human intervention and then use statistical machine learning to fill the remaining steps. With the help of pre-defined priors (e.g., rules, modelling) from external experts, Neuro-Symbolic model introduces a more explainable, controllable and efficient decision-making process [Mao et al., 2021].

Interestingly, Such an approach of modelling a symbolic representation of the world using some expert priors in Neuro-Symbolic AI is closely related to model-based RL [Hafner et al., 2020, Nagabandi et al., 2018, Zhang et al., 2019, Liu et al., 2022, Ecoffet et al., 2019], where agents also construct a predictive model of the world. With appropriate forward-looking exploration strategy, model-based approach could reach the performance comparable to the model-free RL but with fewer interactions. Specifically, there are two ways of deriving solutions based on the world model:

1. Solutions are derived by open-loop re-planning on that world model until the real goal state is reached. For example, Liu et al. [2022] designed a model-based RL agent to solve "instruction following task". The agent used a spatial graph to represent dynamics and then conduct extensive exploration to calibrate the semantic spatial graph. Meanwhile, the natural language instruction was parsed into a planning goal. After that , a planner will treat the graph as the state of the world and conduct Goal Oriented Action Planning (GOAP) [Orkin, 2005] to find a sequence of actions towards the goal state. Further exploration process will be conducted if the plan failed. The empirical results show that this approach could achieve higher success rate than previous RL methods on unseen environments in ALFRED benchmark [Shridhar et al., 2020a,b].

2. Through Actor-Critic learning, a Critic model will be constructed to estimates discounted total future rewards $V^{\pi}(s_t) = \sum_{t'=t}^{T} E_{\pi_\theta}[r(s_{t'}, a_{t'})|s_t]$ that the current Actor model π_θ can achieve. The world model will predict the next state $\hat{s}_{t+1}$ from a given action a and the Actor model π_θ is trained to output actions that lead to the state with highest sum of future rewards estimated by the Critic model $\theta^\star = \arg\max_\theta E_{s_{t+1}\sim\pi_\theta(a_t|s_t)p(s_{t+1}|s_t,a_t)}[V^{\pi_\theta}(s_{t+1})]$. This approach is also known as "model-based acceleration" for model-free RL [Hafner et al., 2020, Janner et al., 2019].

Essentially, the research outcomes from both Neuro-Symbolic and Model-based RL approach imply that modelling of the world (either approximation in the latent space or direct symbolic representation) show the potential of improving sample efficiency and generalisation. This will certainly bring some insight into our research.

2.4.3 Semantic Parsing and Planning

Our task can be seen as to disambiguate natural language strategies or instructions into concrete sequences of actions that transit the state of the game environment. Semantic parsing, on the other hand, involves mapping natural language utterances into logical forms that can be executed by a program. So, to some extent we could describe our task as a semantic parsing problem. Compared to other Natural Language Processing (NLP) tasks, semantic parsing focuses more on how to extract the semantic representation of natural language into formal language. Applications of semantic parsing include question answering, dialogue system and code generation etc. Although NLP models tend to be quite homogeneous nowadays (everyone prefers deep learning Transformer models), recent studies on semantic parsing attempt to promote performance through a divide-and-conquer strategy [Berant and Liang, 2014, Pasupat and Liang, 2016, Dong and Lapata, 2018, Shin et al., 2021]. Specifically, they attempted to parse the natural language into some kinds of intermediate coarse products first (e.g., canonical natural language utterance, macro grammars) and then continue to parse the semi-finished products into desired logical forms. The reason behind the enhanced performance is because the expert intervention at the intermediate layer provides extra guidance for semantic parser to learn.

The principle of “divide and conquer” has been also applied to the study of classical planning on problems informed in natural language. Researchers have attempted to parse natural language information (e.g., instruction manual) into Planning Domain Definition Language (PDDL) first – a machine understandable language – before feeding the information into the AI solver [Lindsay et al., 2017, Kim et al., 2017, Miglani and Yorke-Smith, 2020].

In fact, parsing problems stated in natural language format into machine understandable format is hard because some crucial information may be dropped unknowingly during the transformation. For example, Patel et al. [2019] found out that problem stated in natural language often contains temporal constraints, but temporal constraints are not expressible by Markov rewards. Therefore, we are losing some constraints of the problem when it is transformed into Markov Decision Process formulation. This phenomenon about the expressivity of Markov reward was also discussed by Abel et al. [2021]. To solve this issue, Vaezipoor et al. [2021] suggested to use Linear Temporal Logic (LTL) to encode the temporal constraints in the natural language instructions and also design a non-Markovian (i.e., interaction at previous steps may affect the system at a later time) reward function that only provide rewards if the trajectory align with the instructions expressed as LTL (reward shaping was used here).

There is also one interesting work from Iyer et al. [2019], which suggested that turning common logical form snippets into macro will significantly reduce training time by more than 50%. This finding will be taken into consideration when we move to the harder learning environment NetHack, which consists of a unprecedentedly large action space.

2.5 Data augmentation and Pre-training

Data augmentation and pre-training are the two model-agnostic techniques which have shown significant effectiveness in improving models' performance in other machine learning domains. Yet, only a few attempts have been made to apply these two techniques to RL models, and it remains unclear how to do it effectively. Among those attempts, Goldwasser and Roth [2014] and Akyürek and Andreas [2022] proposed a way of increasing labeled demonstration data for self-supervised learning by augmenting object attributes in the game environment. Meanwhile, Reid et al. [2022] examined the transferability of a Transformer-based language model pre-trained using English Wikipedia Corpus. They copied the parameters of the pre-trained language model into a Transformer-based RL model (i.e., Decision Transformer [Chen et al., 2021]) and the results showed that pre-training weights consistently improved efficiency and decision-making performance.

Chapter 3

Proposed Methods

In this chapter, we propose several methods to answer our research questions. Recall that we will examine our ideas about language-assisted RL mostly in reward shaping paradigm.

Some research questions stated in Section 1.1 may not yet be covered. The remaining research questions will be addressed as we develop. Also, some of the proposed methods will be refined and improved as the project progresses.

3.1 State-Aware Language-Assisted Reward Shaping

Goyal et al. [2019]'s reward shaping module (see Section 2.1.1) only considered to align natural language instructions with kinaesthetic data from the agent trajectory. Consequently, the model is not able to provide accurate rewards when the instruction contains constraints about the state. For instance, if an instruction says "jump over the skull", the reward module has nowhere to perceive the skull and so agents may receive rewards simply by a standing jump. Therefore, we propose to extend their model by taking the visual modality (i.e., the observation data) into account, which requires multimodal fusion. We will stick to contrastive learning method to train the module – to be more specific, one way of doing it is to first of all generate negative samples where action-text model think the sample is coherent, but actually the vision shows it's not (i.e., standing jump, no skull), after that, we train the reward shaping module to correct identify the incoherence.

Meanwhile, we also propose the following methods to improve the performance of the model:

3.1.1 Self attention of frames in each trajectory clips

As mentioned in Section 2.3.3, Transformer-based model is not sensitive to the temporal logic in the data. However, understanding the temporal logic is essential when predicting whether the trajectory and the instruction are coherent. As mentioned in Section 2.4.3, natural language instructions often contains temporal constraints (e.g., “eat food after a battle”, although “eat food” appears in the front of the sentence, the right temporal order of “eat food” is behind the “battle” event). So, leaving all the temporal logic understanding work to the Transformer-based reward shaping module is not recommended. Therefore, we propose to train the visual encoder based on self-attention (i.e., intra-cross attention) among frames in the same episode. By having attention mechanism across episodic frames, we hope that the visual encoder is able to output the latent vector that contains the temporal logic information of the scene. The temporal-aware latent vector will be passed to the reward shaping module, which hence may help to reduce the burden of reward shaping module in handling the temporal logic.

The visual encoder will be trained under Masked Auto-Encoding (MAE) He et al. [2022] manner.

3.1.2 Soft-discretisation of visual inputs

Reasoning can be defined as follows: given a set of atomic evidences, reasoning defines a process where atomic evidences are composed, possibly through a hierarchical step, to form a more abstract concept. Therefore, identifying whether agent behaviours are aligned with instructions relies on the reasoning skill – an agent has to gather information from each time step in the trajectory and then goes through the process of inductive reasoning to decide whether the trajectory matches the abstract language instruction. As mentioned in Section 2.4.2, a pure neural network model is not able to perform reasoning quite well. Therefore, in order to enhance the reasoning ability of the reward shaping module, we propose to apply “soft-discretisation” on the raw input. “Discretisation” is one way of transforming continuous data modality into discrete forms. Given that reasoning only applies to atomic evidences, “discretisation” process becomes essential for models that takes continuous raw data as input to fit in the reasoning paradigm. For example, assuming we want to construct a deep learning model that can perform physical reasoning about the scene. Also given the fact that the atomic unit of Newtonian physical interactions is object, it is then a reasonable pre-process to convert visual data from pixel-based into object-based modality. By doing this, we essentially encourage the model to purely focus on the single physical reasoning task, which therefore causes the model to converge faster. Given the fact that objects are the atomic unit for game interaction and game walkthrough describes events by referring to objects instead of pixels, we propose to add an additional discretisation process to convert pixel-based visual data into object-based before we feeding them into the reward shaping module. Specifically, we want to use a object recognition model to pre-process the pixel data. After that we extract the latent representations from the object recognition model and feed them into the reward shaping network.

3.1.3 Lightweight multimodal fusion

The reward shaping module is required to fuse different unimodal representation of data (visual, language and kinaesthetic) into a unified multimodal representation. Recall that Attention-based fusion method may exacerbate sample inefficiency of RL model as they are data-hungry [Hessel and Lee, 2020]. Therefore, given the fact that Goyal et al. [2019]'s training dataset only contains about 5000 number of entries, we propose to use a lightweight multimodal fusion method and then investigate if there will be performance increase. Low Rank Fusion [Sahay et al., 2020] will be considered as it is a lightweight tensor fusion method.

3.1.4 Parameters transfer from pre-trained language model

As mentioned in Section 2.5, using a large-scale pre-trained language model as the base and fine-tune it on downstream tasks is able to alleviate the data shortage of the downstream task. Since Goyal et al. [2019]'s dataset only contains about 5000 number of data entries, which is not adequate to train a heavy language model such as Transformer, we propose to apply the knowledge transfer technique to facilitate the model. Strictly speaking, the reward shaping module is not a language model but a multimodal model. So, transferring parameters from a pure language model directly to a multimodal model may not be effective. Therefore, we plan to use gated residual connection, inspired by the structure from Alayrac et al. [2022] to smoothly merge the parameters from the pre-trained language model. Besides that, we will replace the current global word embedding, GloVe [Pennington et al., 2014], to contextualised word embedding, BERT, so as to better capture the semantic meaning of each word.

3.1.5 Data augmentation to generate more training data

Goyal et al. [2019]'s dataset only contains around 5000 instruction-demonstration pairs, which is often not adequate for training a deep learning model. Fortunately, our learning environment is virtual video game, which allows us to collect and augment game-play data in a much more convenient way compared to real world learning environment. Nevertheless, collecting human game play data may still be expensive. Therefore, inspired by Goldwasser and Roth [2014] and Akyürek and Andreas [2022], we propose to create an automatic data augmentation method that can generate more and diverse annotated training data. Specifically, given an language and demonstration data pair form the existing dataset, we first of all try to decompose the language instruction into conceptual terms and then augment it with the other values in the same class. For instance, in NetHack environment we have a demonstration and instruction pair where the instruction says "go top to fight with a yellow goblin". We then can extract the terms that can be augmented – "top", "yellow", "goblin". After that, we change the them to other values in the same class, such as "top" → "bottom" (direction class), "yellow" → "red" (colour class), "goblin" → "zombie" (monster class). We subsequently generate the corresponding demonstration "go bottom to fight with a red zombie" in the game environment. By doing this, we just create a new data pair for training.

This work will address both $\boldsymbol{RQ_{1a}}$ and $\boldsymbol{RQ_{1b}}$.

3.2 Translating Agent Behaviour using Connectionist Temporal Classification

We propose using Connectionist Temporal Classification (CTC) to translate the expert agent behaviour into natural language walkthrough. The advantage of using CTC is that we do not need to provide the exact alignment labels between the input and the output sequences. However, the demonstrations and the corresponding natural language walkthrough have to satisfy the three CTC properties:

1. the alignment of demonstrations to walkthrough text sequences are monotonic;
2. the alignment of demonstrations to walkthrough text is "many-to-one";
3. the length of demonstration sequences must be greater than the length of walkthrough text, nevertheless the third property can be inferred from the preceding two.

The challenge of this approach is that if we want to use supervised learning, we have to ensure that the training data pairs obey the first property. Therefore, we propose that we can parse the natural language walkthrough into corresponding Linear Temporal Logic (LTL) expression first, and then based on the LTL expression, we rewrite the walkthrough into a coherent sequential order so that the first property is satisfied (this approach will link back to the proposed method in Section 3.4).

This work is still developing and will address $\boldsymbol{RQ}_{2b}$. This work is the prerequisite of the proposed method in Section 3.3.

3.3 Hindsight Experience Relabelling using Natural Language

The core idea behind Hindsight Experience Relabelling (HER) [Andrychowicz et al., 2017] is that when the agent fails to achieve the goal at the current round, we manually modify the goal so that the previously undesirable actions will become desirable. As such we will have more positive experiences (it is almost impossible for an agent to naturally obtain a positive experience for complex tasks). Packer et al. [2021] further attempted to relabel the task instead of the goal under the same principle so as to generalise HER to multi-task environment.

The goal of our study is to use task-specific natural language information to facilitate learning of game-playing agents. Learning from demonstration (i.e., Imitation Learning) is one common approach. However, our proposed model suffers from the shortage of game-play demonstration data. We could create game-play demonstration data by recording expert players playing the game, but such an approach is very expensive. Inspired by some previous studies, we could try to collect demonstration data by using a baseline game-playing agent (or novice humans) to interact with environment. However, the issue of this approach is that, most of the demonstrations are negative because the game is too difficult for a baseline agent to play. It turns out that the agent may not be able to learn effectively if the demonstrations are imbalanced.

Inspired by work from Packer et al. [2021], we might be able to utilise the mainly bad demonstrations. What we need to do is to just annotate those imbalanced replays using natural language – meaning that we just describe the what happened in the replay no matter if the replay is positive or not. So, what we are doing there can be seen as re-assigning a different task $\mathcal{T}'$ to the replay data descried in natural language $L(\mathcal{T}')$. And it turns out that although the replay is not a good demonstration for the original game $\mathcal{T}$, it is a good demonstration for the task $\mathcal{T}'$. By conducting the HER techniques, we expect the agent to learn how to play the game even from sparse positive reward signals. Specifically HER was done as follows:

1. Obtain one trajectory (episode) from the replay buffer. A trajectory is defined as a sequence of transition $\{s_t, a_t, r(s_t, a_t, \mathcal{T})\}$ from time step 0 until a terminal state is reached, where s_t is the state, a_t is the action, $r(s_t, a_t, \mathcal{T})$ is the immediate reward which is determined by the current state, action and the current task $\mathcal{T}$, and $r(s_t, a_t, \mathcal{T})$ for most t is 0 because task $\mathcal{T}$ is difficult;

2. Rewrite the transition $\{s_t, a_t, r(s_t, a_t, \mathcal{T})\}$ into $\{s_t, a_t, r(s_t, a_t, \mathcal{T}_{\text{perfect}})\}$, where $\mathcal{T}_{\text{perfect}}$ is an alternative task that provide every transition with positive feedback – meaning $r(s_t, a_t, \mathcal{T}_{\text{perfect}}) = +1$ forall t In our context, this means that $L(\mathcal{T}_{\text{perfect}})$ is just the description of the toxic trajectory. Since $L(\mathcal{T})$ is the language representation of $\mathcal{T}$, we can replace all $\mathcal{T}$ with $L(\mathcal{T})$ without breaking any conditions. $L(\mathcal{T}_{\text{perfect}})$ can be obtained using proposed method in Section 3.2;

3. $\mathcal{T}_{\text{perfect}}$ is not yet a good trajectory for training, because there is no bad moves and therefore RL agent trained by this transition will become over-confident. At this stage, we need to derive a new task $\mathcal{T}'$ so that the $\{r(s_t, a_t, \mathcal{T}')\}$ of this trajectory is a balanced collection of positive and negative feedback. We can derive $\mathcal{T}'$ and $\{r(s_t, a_t, \mathcal{T}')\}$ by human labelling from Amazon Mechanical Turk. After that, we append the hindsight trajectory into hindsight replay buffer.

4. Repeat the previous steps until getting enough size of hindsight replay buffer. Finally, train the RL agent using the hindsight replay buffer.

Here is a justification why HER may work for language-assisted learning agents: toxic demonstrations can be utilised by understanding the compositionality in tasks with the help of natural language descriptions. For example, we have a **toxic** demonstration about an agent who jumped into a snake and died. This demonstration can then be decomposed in two dimensions – 1). objects “snake” and 2). action “jump into”. The agent turns out to learn two **neutral** knowledge through the decomposition process. After extensive training, we provide the agent with the correct instruction ”you should jump over a snake” and the agent is expected to composite knowledge of “snake” and knowledge of “jump over” together to form a **desirable** action plan.

As seen in some semantic parsing work mentioned in Section 2.4.3, some simple annotations from experts could make a significant acceleration to the learning process. Therefore, in addition to the above proposed method, which contains using human labelling from Amazon Mechanical Turk, we propose to further tag game walkthroughs as “recommendations” or “constraints”, so that the reward shaping module will not only reward but also punish the agents if their behaviours matches the constraints. The measurement will be based on some similarity metrics such as cosine similarity between two latent vectors.

This work is still developing and will address $\boldsymbol{RQ}_{1b}$.

3.4 Linear Temporal Logic (LTL) to Extract the Temporal Ordering Information in Natural Language

As mentioned in Section 2.4.3 and Section 2.3.3, natural language often contains temporal constraints, but temporal constraints are often not captured by Transformer-based model. Therefore, it is worth trying different reward shaping module instead of the transformer-based one like Goyal et al. [2019]'s so as to produce more accurate shaping rewards. Inspired by the studies from Vaezipoor et al. [2021] and Patel et al. [2019], we will examine if it is better to introduce LTL expression into the reward shaping process.

Recall that LTL is the extension of propositional logic with two extra temporal operators: $\bigcirc$ (next) and $\cup$ (until). Given a finite set of propositional symbols $\mathcal{P}$, the synatx of LTL formula is defined as $\varphi ::= p|\neg\varphi|\varphi \wedge \psi| \bigcirc \varphi|\varphi \cup \psi$, where $p \in \mathcal{P}$. Compared with propositional logic, LTL formulas are evaluated over sequences of observations – meaning that we have time step i. Intuitively, the formula (next φ) holds if φ holds at the next time step and (φ until ψ) holds if φ holds until ψ holds. In addition to that, we can also define Boolean operators $\vee$ (or) and $\rightarrow$ (implication), and the temporal operator $\Box$ (always), $\Diamond$ (eventually). By using LTL, we are able to precisely express the temporal logic of events. For more formal definition, please check the textbook.

So, we will convert natural language instructions into LTL expression first and then align the LTL expression with the trajectory using Neuro-Symbolic method such as Graph Neural Network (GNN) so that we can avoid the use of Transformer but still trying to capture the temporal ordering of events. By doing this, we expect that the language reward shaping module becomes more sensitive to the temporal relationship of events mentioned in the natural language data. Another contribution of this experiment is to find an effective way of parsing natural language into LTL expression.

This work is still developing and will address $\boldsymbol{RQ}_{1a}$.

3.5 Longer Sequences of Game-Play Clips

In order to examine the capacity of the existing language reward shaping method, we could independently annotate longer sequences of game-play from the Goyal et al. [2019] dataset, and compare the longer and complex language instructions versus the short 3-second data from Goyal et al. [2019]. This might help to understand a suite of issues in using more natural instructions, versus explicit low-level walkthroughs. We propose the following method to deal with longer and more natural instructions:

As mentioned in Section 2.4.1, dialogue system often encounter a long sequence of decision-making and the information and some correct decisions can only be made by referring to agents' past experiences. In order to deal with such a "looking back at history" situation, dialogue systems often explicitly model an additional memory model that was used to construct a latent variable that integrates agent's observations over time. Therefore, we also plan to construct an extra episodic memory module so as to deal with long trajectories. Also, in order to handle information in abstract modality, we plan to use Graph Neural Network – a Neuro-Symbolic model to construct our memory model so that we can manipulate the topology of knowledge. After that, in terms of handling long natural language data, we will apply "divide and conquer" principle and try to distinguish "abstract strategies" and "step by step instructions" from the text source. The separation method can be based on uncertainty of the information. After splitting the text data into separate parts, we will try to use Option MDP to construct the agent policy model. More details about Option MDP is mentioned in Section 2.2.1.

This work will address $\boldsymbol{RQ}_{2a}$.

3.6 Segmentation of Long Walkthrough and Demonstration Snippets Retrieval

Inspired by retrieval-based Question Answering (QA) system, we propose a pipeline for game-playing agent to handle long game walkthrough data. Here are the steps:

1. Collect instruction-demonstration short clips into a database. Data augmentation method mentioned in Section 3.1.5 can be used here.
2. Given a long walkthrough, segment it into sub-goals. This can be done by Option MDP mentioned in Section 3.5.
3. Based on the temporal order, use each sub-goal as query to retrieve relevant demonstration clips from the the database. The relevancy can be measured based on vector cosine similarity, assuming we have a trained similarity function, which we will have one once we complete the work in Section 3.3.
4. Construct an action plan based on the selected relevant demonstrations. Execute the plan accordingly. We can either use Reinforcement learning to find the action plan or we conduct a open-loop re-planning method mentioned in Section 2.4.2.

5. Detect whether each sub-goal is completed – for learning based agent, train a sub-goal completion detection model similar as training the reward shaping module mentioned in Section 2.1.1; for planning based agent, we build a world model for the planner, re-calibrate the world model and re-plan if the agent encountered unexpected circumstances or the plan cannot reach the goal state.

6. Repeat 3-5 steps until all sub-goals are fulfilled.

This work is considered as the integration of all the previous proposed method so as to establish a complete system of language assisted game-playing agent. This work is still developing.

Chapter 4

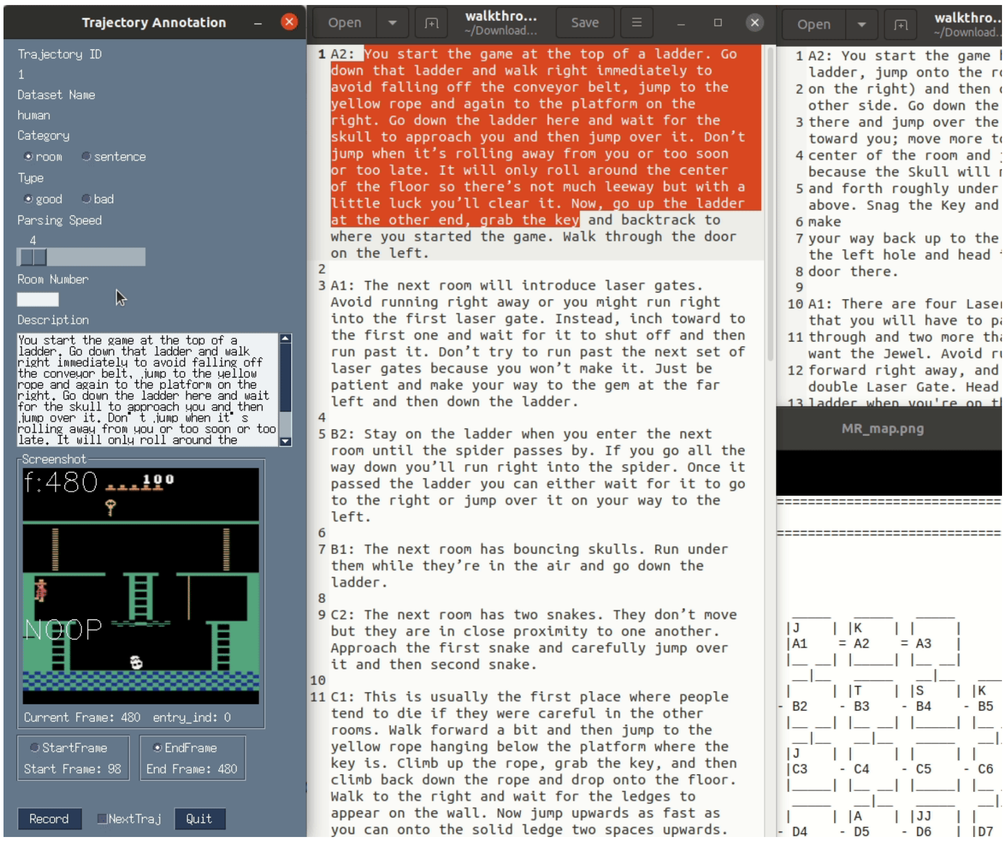
Progress to Date

We spent a long period of time on the literature review, and also conducted several time-consuming preliminary tests in order to find out the best game environments for our research, which should have suitable difficulty level, adequate language data and no bugs (we have considered Angry Birds [Xue et al., 2022] and IGLU environment [Kiseleva et al., 2022] but eventually we decided to use Montezuma's Revenge [Machado et al., 2018] and NetHack [Küttler et al., 2020]). Other than that, I took about 3 months (20-Feb-2022 to 14-May-2022) conducting experiments on the NetHack environment and then it appeared that NetHack game is too complicated for learning-based agent to deal with so eventually I moved my attention to Montezuma's Revenge game, which has a suitable difficulty level to be the starting point.

So, unfortunately, the expectations listed in pre-confirmation report were not met because the previous learning environment IGLU was buggy and thus we decided to abandon it as well as the associated old research proposal. Nevertheless, we are now in the process of developing our first proposed method outlined in Section 3.1 and we are quite optimistic about our new direction as well as the progress.

4.1 Data Annotation

The proposed methods in Section 3.1, Section 3.3 and Section 3.5 all require having further annotated dataset. So, an annotation tool has been made to annotate trajectories (See Figure 4.1).



```
Trajectory ID
1
Dataset Name
human
Category
(•) room   ( ) sentence
Type
(•) good   ( ) bad
Parsing Speed
4
Room Number

Description
You start the game at the top of a
ladder. Go down that ladder and walk
right immediately to avoid falling off
the conveyor belt, jump to the yellow
rope and again to the platform on the
right. Go down the ladder here and wait
for the skull to approach you and then
jump over it. Don't jump when it's
rolling away from you or too soon or too
late. It will only roll around the
Screenshot
f:480
NOOP
Current Frame: 480   entry_ind: 0
( ) StartFrame          (•) EndFrame
Start Frame: 98         End Frame: 480
```

```
A2: You start the game at the top of a ladder. Go
down that ladder and walk right immediately to
avoid falling off the conveyor belt, jump to the
yellow rope and again to the platform on the
right. Go down the ladder here and wait for the
skull to approach you and then jump over it. Don't
jump when it's rolling away from you or too soon
or too late. It will only roll around the center
of the floor so there's not much leeway but with a
little luck you'll clear it. Now, go up the ladder
at the other end, grab the key and backtrack to
where you started the game. Walk through the door
on the left.

A1: The next room will introduce laser gates.
Avoid running right away or you might run right
into the first laser gate. Instead, inch toward to
the first one and wait for it to shut off and then
run past it. Don't try to run past the next set of
laser gates because you won't make it. Just be
patient and make your way to the gem at the far
left and then down the ladder.

B2: Stay on the ladder when you enter the next
room until the spider passes by. If you go all the
way down you'll run right into the spider. Once it
passed the ladder you can either wait for it to go
to the right or jump over it on your way to the
left.

B1: The next room has bouncing skulls. Run under
them while they're in the air and go down the
ladder.

C2: The next room has two snakes. They don't move
but they are in close proximity to one another.
Approach the first snake and carefully jump over
it and then second snake.

C1: This is usually the first place where people
tend to die if they were careful in the other
rooms. Walk forward a bit and then jump to the
yellow rope hanging below the platform where the
key is. Climb up the rope, grab the key, and then
climb back down the rope and drop onto the floor.
Walk to the right and wait for the ledges to
appear on the wall. Now jump upwards as fast as
you can onto the solid ledge two spaces upwards.
```

```
 1 A2: You start the game
   ladder, jump onto the r
 2 on the right) and then
   other side. Go down the
 3 there and jump over the
   toward you; move more t
 4 center of the room and
   because the Skull will
 5 and forth roughly under
   above. Snag the Key and
 6 make
 7 your way back up to the
   the left hole and head
 8 door there.
 9
10 A1: There are four Lase
   that you will have to p
11 through and two more th
   want the Jewel. Avoid r
12 forward right away, and
   double Laser Gate. Head
13 ladder when you're on t
```

```
=============================

=============================

  _____   _____   _____
 |J     | |K    | |     |
 |A1     = A2    = A3   |
 |__ __| |_____| |__ __|
  __|__   _____   __|__   ___
 |      | |T    | |S    | |K
- B2    - B3    - B4    - B5
 |__ __| |__ __| |_____| |__
  __|__   __|__   _____   __|
 |J     | |     | |     | |
 |C3    - C4    - C5    - C6
 |_____| |__ __| |_____| |__
  _____   __|__   _____   __|
 |      | |A    | |JJ   | |
- D4    - D5    - D6    | |D7
```

FIGURE 4.1: A annotation tool to annotate Montezuma's Revenge demonstrations

4.2 State-Aware Language-Assisted Reward Shaping

Several preliminary preparation work has been done in order to set up the environment for the proposed state-aware language-assisted reward shaping method mentioned in Section 3.1 (See Table 4.1).

No.	Preparation	Date
1	Collect demonstration and walkthrough dataset	22-May to 28-May
2	Align demonstration and walkthrough	29-May to 04-Jun
3	Literature review on multimodal alignment	05-Jun to 18-Jun
4	Rebuild Goyal et al. [2019]'s model using PyTorch[Paszke et al., 2019] and Nvidia DALI	12-Jun to 25-Jun

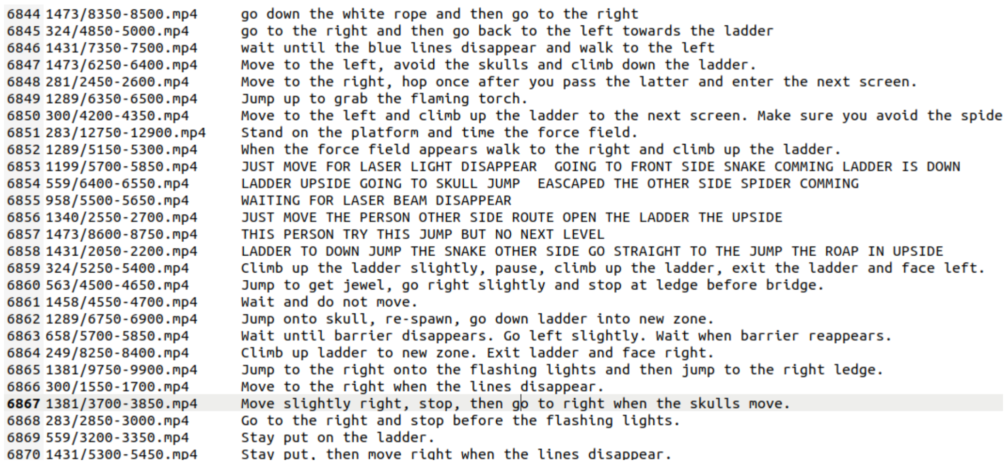
TABLE 4.1: Preparation work summary

4.2.1 Experiment setup

4.2.1.1 Training data

To train and evaluate the state-aware reward shaping model, we need to collect both the trajectory dataset and the annotated trajectory dataset. Trajectory data of Montezuma's Revenge game is collected from Atari Grand Challenge dataset, which contains about 1 million frames of human player demonstrations. However, most of the demonstrations are from novice players who did not even escape from the first room.

The annotated trajectory dataset comes from Goyal et al. [2019]'s work. It contains about 5000 annotated 3-second demonstration clips as training data and 3000 as test data. Annotations are one simple sentence that describe the event of the 3-second demonstration (See Figure 4.2).



```
6844 1473/8350-8500.mp4     go down the white rope and then go to the right
6845 324/4850-5000.mp4      go to the right and then go back to the left towards the ladder
6846 1431/7350-7500.mp4     wait until the blue lines disappear and walk to the left
6847 1473/6250-6400.mp4     Move to the left, avoid the skulls and climb down the ladder.
6848 281/2450-2600.mp4      Move to the right, hop once after you pass the latter and enter the next screen.
6849 1289/6350-6500.mp4     Jump up to grab the flaming torch.
6850 300/4200-4350.mp4      Move to the left and climb up the ladder to the next screen. Make sure you avoid the spide
6851 283/12750-12900.mp4    Stand on the platform and time the force field.
6852 1289/5150-5300.mp4     When the force field appears walk to the right and climb up the ladder.
6853 1199/5700-5850.mp4     JUST MOVE FOR LASER LIGHT DISAPPEAR  GOING TO FRONT SIDE SNAKE COMMING LADDER IS DOWN
6854 559/6400-6550.mp4      LADDER UPSIDE GOING TO SKULL JUMP  EASCAPED THE OTHER SIDE SPIDER COMMING
6855 958/5500-5650.mp4      WAITING FOR LASER BEAM DISAPPEAR
6856 1340/2550-2700.mp4     JUST MOVE THE PERSON OTHER SIDE ROUTE OPEN THE LADDER THE UPSIDE
6857 1473/8600-8750.mp4     THIS PERSON TRY THIS JUMP BUT NO NEXT LEVEL
6858 1431/2050-2200.mp4     LADDER TO DOWN JUMP THE SNAKE OTHER SIDE GO STRAIGHT TO THE JUMP THE ROAP IN UPSIDE
6859 324/5250-5400.mp4      Climb up the ladder slightly, pause, climb up the ladder, exit the ladder and face left.
6860 563/4500-4650.mp4      Jump to get jewel, go right slightly and stop at ledge before bridge.
6861 1458/4550-4700.mp4     Wait and do not move.
6862 1289/6750-6900.mp4     Jump onto skull, re-spawn, go down ladder into new zone.
6863 658/5700-5850.mp4      Wait until barrier disappears. Go left slightly. Wait when barrier reappears.
6864 249/8250-8400.mp4      Climb up ladder to new zone. Exit ladder and face right.
6865 1381/9750-9900.mp4     Jump to the right onto the flashing lights and then jump to the right ledge.
6866 300/1550-1700.mp4      Move to the right when the lines disappear.
6867 1381/3700-3850.mp4     Move slightly right, stop, then go to right when the skulls move.
6868 283/2850-3000.mp4      Go to the right and stop before the flashing lights.
6869 559/3200-3350.mp4      Stay put on the ladder.
6870 1431/5300-5450.mp4     Stay put, then move right when the lines disappear.
```

FIGURE 4.2: Short and simple annotation from Goyal et al. [2019] dataset

We will also use trajectory dataset generated from DQN agent from Google DQN replay dataset. However, the trajectory from DQN agent is very noisy (See the comparison Table 4.2).

4.2.1.2 Implementation details

"Free model upgrade"

After building the PyTorch version Goyal et al. [2019]'s reward shaping model as the baseline model, I continued to modify this baseline model by upgrading its model components to State-of-the-art (SOTA) (e.g., from fully-connected layer network backbone to Transformer network backbone). These updates are not even mentioned in our proposed method section. Nevertheless, I am doing this because I want to fully examine the potential of the state-aware language reward shaping method (See Table 4.3).

	DQN replay dataset	**Human replay dataset**
image	84 x 84 grayscale	160 x 210 x 3 rgb
actions	we cannot really control the character when he is in the air, and we can see that DQN agent is not aware of it. Also, the action sequence of the agent looks disordered (So, if we want to utilise the trajectory to train agent, it is better to clean the noises in the action sequence and extract the valid action)	the action is very consistent and steady compared to DQN agent
contrastive learning	Most of them can be seen as bad experiences which teach agent what shouldn't do.	Most of them are bad experiences too

TABLE 4.2: Comparison table about human player dataset and DQN dataset

	Upgrade	**Reason**
1	Change the format of action data from bag (multiset) of actions into sequential patches of smaller bag of actions (10 frames per patch).	encode temporal information
2	Change fully-connected layer to a Transformer variant from Alayrac et al. [2022]	a better model for temporal sequential data
3	Change Transformer to Norm-Former[Shleifer et al., 2021] + RM-SNorm[Zhang and Sennrich, 2019]	more stable convergence

TABLE 4.3: What components of the reward shaping module were upgraded as well as the reason

After that, I pre-trained a visual encoder using MAE variant manner [Seo et al., 2022] so that I can take the latent vector of the observation data into the language reward shaping module. It was trained 40 epochs on Montezuma human replay dataset ($\sim$ 3 days).

4.2.1.3 Results

Since the reward shaping model is trained based on contrastive learning, negative examples are created by 1) sampling action-language pair from a given trajectory, but choosing image data from alternative trajectory (i.e., Negative image example) 2) sampling action-image pair from a given trajectory, but choosing language data from alternative trajectory (i.e., Negative language example) 3) sampling image-language pair from a given trajectory, but generate a random action sequence based on uniform distribution (i.e., Negative action example).

The reward shaping model performance was evaluated by calculating the rate of correct prediction of whether the input trajectory and language instruction are matched (i.e., accuracy). The model was tested on the Goyal et al. [2019]'s test data that consists of 4000 paired data.

The result of state-aware reward shaping model and its variants based on how to arrange q, k, v input of the Transformer layer are shown in Table 4.4.

	Model	**Accuracy**
1	Baseline (without image)	85.08%
2	Upgraded Baseline (without image)	89.17%
3	q = text, k = v = concat(image, action)	62.04% @ epoch 88
4	q = text, k = v = concat(reduce(image), action)	73.62% @ epoch 147
5	q = text, k = reduce(image), v = action	71.19% @ epoch 25
6	q = text, k = v = concat(reduce(image), amplify(action))	72.40% @ epoch 75
7	q = text, k = reduce(image), v = amplify(action)	72.24% @ epoch 21

TABLE 4.4: Performance of the model on different q, k, v setting of the Transformer layer. reduce and amplify function are fully-connected layer that changes the feature dim of the input before feeding into the transformer.

Confusion matrix was created to help to analyse the performance (See Figure 4.3).

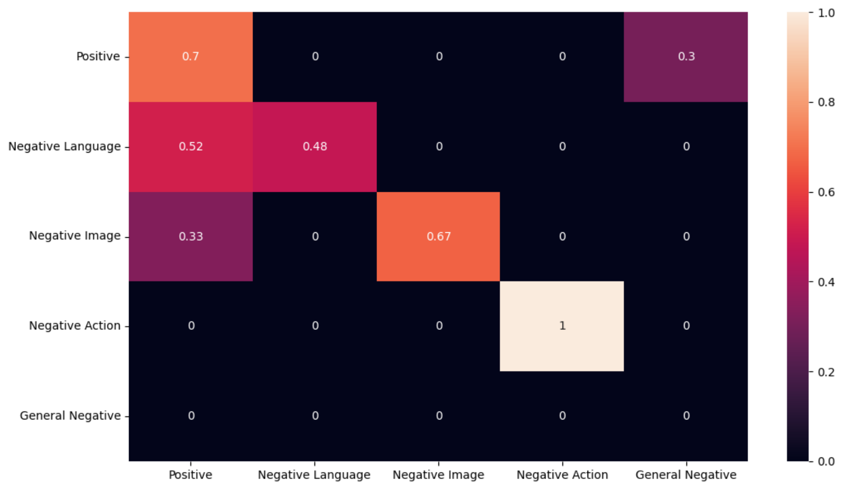


FIGURE 4.3: Confusion matrix of the multimodal reward shaping module, we can see that the False Positive value of predicting language

There are some observations regarding to the modified Goyal et al. [2019]'s model:

1. Precision-recall trade off happened:

- as the training process goes on, the model's precision increases but recall decreases (meaning that the network tends to become more conservative to give reward signals);
- it remains unclear about which type of reward signals the RL agent prefer (noisy reward signal versus less sensitive signals);
- Wang et al. [2020] argue that noisy reward signals is good to construct a robust RL agent. But experiments should be conducted to know what type of reward signal suits our circumstance.

2. The model struggled to correctly recognise 1) negative samples where the action-text are paired correctly but the image comes from alternative samples (i.e., negative image sample) and also 2) negative samples where the action-image are paired correctly but the language comes from alternative samples (i.e., negative language sample).

Qualitative evaluation of visual encoder The autoencoder can perfectly reconstruct image from compressed latent vectors (See Figure 4.4).

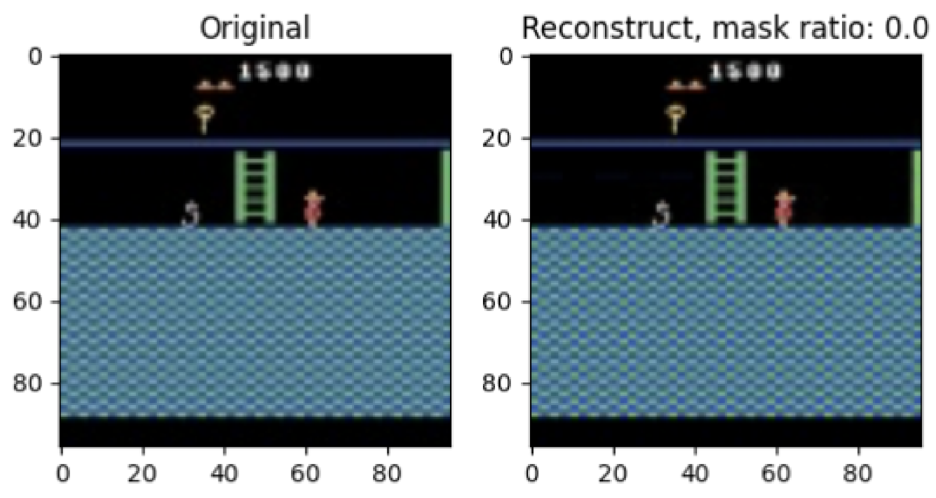


FIGURE 4.4: The autoencoder is able to perfectly reconstruct the image from the latent vector

Interestingly, when the decoder is provided with an empty vector (i.e., mask ratio = 1.0), the decoder will return "first room" scene with no character, no skulls and even no keys (See Figure 4.5). And when the decoder is provided with only 3% of the latent vector (i.e., 0.97 mask ratio), the decoder start to reconstruct the room but the ladder in the room is missed. When 5% of the latent vector is provided (i.e., 0.95 mask ratio), the decoder is able to reconstruct the scene quite well.

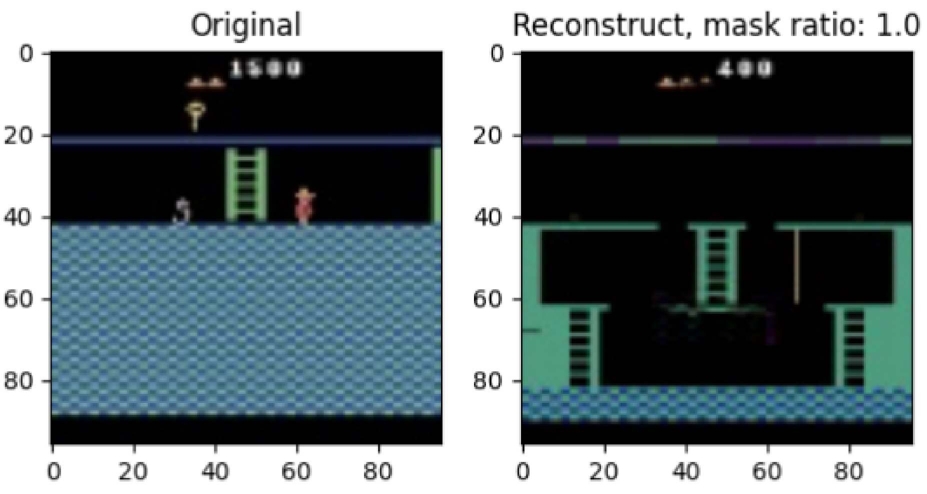
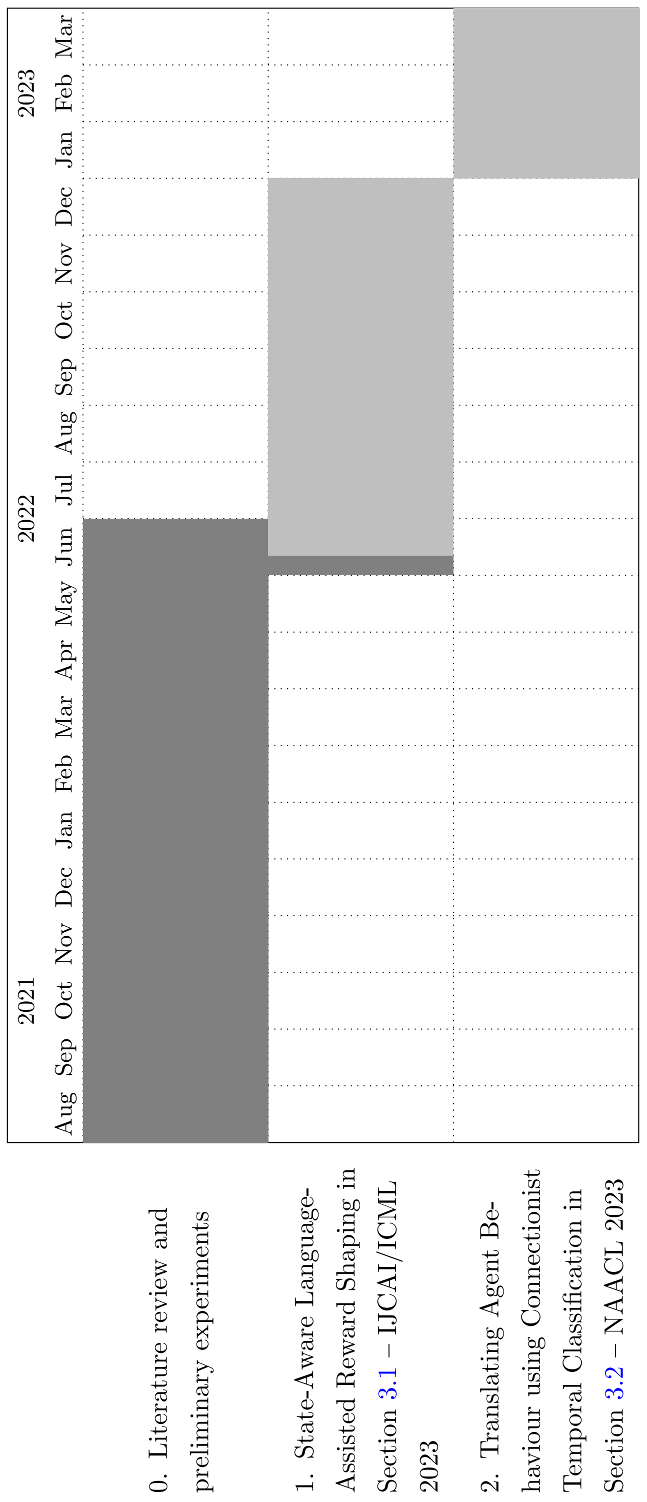


Figure 4.5: Interestingly, when the decoder is provided with an empty vector, the decoder will return "first room" scene with only static objects

The experiment is still underway and please notice all the progress mentioned above is only an intermediate step to the proposed methods in Section 3.1.

Chapter 5

Proposed Schedule and Timeline


2021
2022
2023
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
0. Literature review and preliminary experiments
1. State-Aware Language-Assisted Reward Shaping in Section 3.1 – IJCAI/ICML 2023
2. Translating Agent Behaviour using Connectionist Temporal Classification in Section 3.2 – NAACL 2023

	2023												2024							
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug
3. Hindsight Experience Relabelling using Natural Language in Section 3.3 – ACL / ICAPS 2023				▨	▨	▨	▨	▨	▨	▨										
4. Linear Temporal Logic (LTL) to Extract the Temporal Ordering Information in Natural Language in Section 3.4 – EMNLP 2024											▨	▨	▨	▨	▨					
5. Longer Sequences of Game-Play Clips in Section 3.5 – ACL / ICAPS 2024																▨	▨	▨	▨	▨

	2024				2025							
	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug
5. Longer Sequences of Game-Play Clips in Section 3.5 – ACL / ICAPS 2024	▨	▨										
6. Segmentation of Long Walkthrough and Demonstration Snippets Retrieval in Section 3.6 – EMNLP 2025			▨	▨	▨	▨	▨	▨				
7. Work on thesis									▨	▨	▨	▨

Appendix A

Important dates for each related conference

1. IJCAI (ECAI) – January 7: Abstract submission deadline
2. ICML – Jan 20: Abstract submission deadline
3. NAACL – March 25: Submission deadline for Student Research Workshop papers
4. EMNLP – May 10: Abstract submission deadline (long and short papers); May 17: Submission deadline (long and short papers)
5. NeurIPS – May 22: Abstract submission deadline
6. AAAI – August 30: Abstracts due
7. ACL – November 15: Deadline for submission to ACL Rolling Review
8. ICAPS – December 10: Abstract submission deadline

Bibliography

Alison Gopnik and Andrew Meltzoff. The development of categorization in the second year and its relation to other cognitive and linguistic developments. *Child development*, pages 1523–1531, 1987.

Adrià Puigdomènech Badia, Bilal Piot, Steven Kapturowski, Pablo Sprechmann, Alex Vitvitskyi, Zhaohan Daniel Guo, and Charles Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pages 507–517. PMLR, 2020.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dkebiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Elizabeth S Spelke and Katherine D Kinzler. Core knowledge. *Developmental science*, 10(1):89–96, 2007.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Prasoon Goyal, Scott Niekum, and Raymond J Mooney. Using natural language for reward shaping in reinforcement learning. *arXiv preprint arXiv:1903.02020*, 2019.

Prasoon Goyal, Scott Niekum, and Raymond J Mooney. Pixl2r: Guiding reinforcement learning using natural language by mapping pixels to rewards. *arXiv preprint arXiv:2007.15543*, 2020.

Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning? *arXiv preprint arXiv:2201.12122*, 2022.

Susan Carey and Elsa Bartlett. Acquiring a single new word. *Papers and Reports on Child Language Development*, 15:17–29, 1978.

Bernadette Sharp and Rodolfo Delmonte. *Natural Language Processing and Cognitive Science*. De Gruyter, 2015.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

Chuhan Zhang, Ankush Gupta, and Andrew Zisserman. Temporal query networks for fine-grained video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4486–4496, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Jack Hessel and Lillian Lee. Does my multimodal model learn cross-modal interactions? it's harder to tell than you might think! *arXiv preprint arXiv:2010.06572*, 2020.

Thang M Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks? *arXiv preprint arXiv:2012.15180*, 2020.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. Efficient low-rank multimodal fusion with modality-specific factors. *arXiv preprint arXiv:1806.00064*, 2018.

Julia Kiseleva, Alexey Skrynnik, Artem Zholus, Shrestha Mohanty, Negar Arabzadeh, Marc-Alexandre Côté, Mohammad Aliannejadi, Milagro Teruel, Ziming Li, Mikhail Burtsev, et al. Iglu 2022: Interactive grounded language understanding in a collaborative environment at neurips 2022. *arXiv preprint arXiv:2205.13771*, 2022.

Zhuangdi Zhu, Kaixiang Lin, and Jiayu Zhou. Transfer learning in deep reinforcement learning: A survey. *arXiv preprint arXiv:2009.07888*, 2020.

Eric Hambro, Sharada Mohanty, Dmitrii Babaev, Minwoo Byeon, Dipam Chakraborty, Edward Grefenstette, Minqi Jiang, Jo Daejin, Anssi Kanervisto, Jongmin Kim, et al. Insights from the neurips 2021 nethack challenge. In *NeurIPS 2021 Competitions and Demonstrations Track*, pages 41–52. PMLR, 2022.

Heinrich Küttler, Nantas Nardelli, Alexander Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. The nethack learning environment. *Advances in Neural Information Processing Systems*, 33:7671–7684, 2020.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, pages 2206–2240. PMLR, 2022.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Jacob Andreas, Dan Klein, and Sergey Levine. Modular multitask reinforcement learning with policy sketches. In *International Conference on Machine Learning*, pages 166–175. PMLR, 2017.

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR, 2018.

Lilian Weng. Policy gradient algorithms. *lilianweng.github.io*, 2018. URL https://lilianweng.github.io/posts/2018-04-08-policy-gradient/.

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. *arXiv preprint arXiv:1906.03926*, 2019.

Dimitri Bertsekas. *Dynamic programming and optimal control: Volume I*, volume 1. Athena scientific, 2012.

Hengyuan Hu, Denis Yarats, Qucheng Gong, Yuandong Tian, and Mike Lewis. Hierarchical decision making by generating and following natural language instructions. *arXiv preprint arXiv:1906.00744*, 2019.

Felix Hill, Olivier Tieleman, Tamara von Glehn, Nathaniel Wong, Hamza Merzic, and Stephen Clark. Grounded language learning fast and slow. *arXiv preprint arXiv:2009.01719*, 2020.

Tianshi Cao, Jingkang Wang, Yining Zhang, and Sivabalan Manivasagam. Babyai++: Towards grounded-language learning beyond memorization. *arXiv preprint arXiv:2004.07200*, 2020.

Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, et al. Neurips 2021 competition iglu: Interactive grounded language understanding in a collaborative environment. *arXiv preprint arXiv:2110.06536*, 2021.

Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Dhruv Ramani. A short survey on memory based reinforcement learning. *arXiv preprint arXiv:1904.06736*, 2019.

Hao Hu, Jianing Ye, Guangxiang Zhu, Zhizhou Ren, and Chongjie Zhang. Generalizable episodic memory for deep reinforcement learning. *arXiv preprint arXiv:2103.06469*, 2021.

Marion Blank, Susan A Rose, and Laura J Berlin. *The language of learning: The preschool years*. Grune & Stratton, 1978.

Jens Rasmussen. Information processing and human-machine interaction. *An approach to cognitive engineering*, 1986.

Yiding Jiang, Shixiang Gu, Kevin Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning. *arXiv preprint arXiv:1906.07343*, 2019.

Hannah Bull, Triantafyllos Afouras, Gül Varol, Samuel Albanie, Liliane Momeni, and Andrew Zisserman. Aligning subtitles in sign language videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11552–11561, 2021.

Yang Zhao, Zhou Zhao, Zhu Zhang, and Zhijie Lin. Cascaded prediction network via segment tree for temporal video grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4197–4206, 2021.

Bairong Li, Yuesheng Zhu, Ruixin Liu, and Zhenyu Weng. Learning frame-level affinity with video-level labels for weakly supervised temporal action detection. *Neurocomputing*, 463:109–121, 2021.

Jianan Wang, Boyang Li, Xiangyu Fan, Jing Lin, and Yanwei Fu. Data-efficient alignment of multimodal sequences by aligning gradient updates and internal feature distributions. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 665–675, 2021.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. Augmenting transformers with knn-based composite memory for dialog. *Transactions of the Association for Computational Linguistics*, 9:82–99, 2021.

Young Chol Song, Iftekhar Naim, Abdullah Al Mamun, Kaustubh Kulkarni, Parag Singla, Jiebo Luo, Daniel Gildea, and Henry A Kautz. Unsupervised alignment of actions in video with text descriptions. In *IJCAI*, pages 2025–2031, 2016.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

Cristian-Paul Bara, Sky CH-Wang, and Joyce Chai. Mindcraft: Theory of mind modeling for situated dialogue in collaborative tasks. *arXiv preprint arXiv:2109.06275*, 2021.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, et al. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571, 2020.

Shiquan Yang, Rui Zhang, Sarah Erfani, and Jey Han Lau. An interpretable neuro-symbolic reasoning framework for task-oriented dialogue generation. *arXiv preprint arXiv:2203.05843*, 2022.

David Ding, Felix Hill, Adam Santoro, Malcolm Reynolds, and Matt Botvinick. Attention over learned object embeddings enables complex visual reasoning. *Advances in neural information processing systems*, 34:9112–9124, 2021.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. *arXiv preprint arXiv:1910.01442*, 2019.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

Jiayuan Mao, Haoyue Shi, Jiajun Wu, Roger Levy, and Josh Tenenbaum. Grammar-based grounded lexicon learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

Marvin Zhang, Sharad Vikram, Laura Smith, Pieter Abbeel, Matthew Johnson, and Sergey Levine. Solar: Deep structured representations for model-based reinforcement learning. In *International Conference on Machine Learning*, pages 7444–7453. PMLR, 2019.

Xiaotian Liu, Hector Palacios, and Christian Muise. A planning based neural-symbolic approach for embodied instruction following. *Interactions*, 9:8, 2022.

Adrien Ecoffet, Joost Huizinga, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Go-explore: a new approach for hard-exploration problems. *arXiv preprint arXiv:1901.10995*, 2019.

Jeff Orkin. Agent architecture considerations for real-time planning in games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 1, pages 105–110, 2005.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020a.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020b.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, 2014.

Panupong Pasupat and Percy Liang. Inferring logical forms from denotations. *arXiv preprint arXiv:1606.06900*, 2016.

Li Dong and Mirella Lapata. Coarse-to-fine decoding for neural semantic parsing. *arXiv preprint arXiv:1805.04793*, 2018.

Richard Shin, Christopher H Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. Constrained language models yield few-shot semantic parsers. *arXiv preprint arXiv:2104.08768*, 2021.

Alan Lindsay, Jonathon Read, Joao F Ferreira, Thomas Hayton, Julie Porteous, and Peter Gregory. Framer: Planning models from natural language action descriptions. In *Twenty-Seventh International Conference on Automated Planning and Scheduling*, 2017.

Joseph Kim, Christopher J Banks, and Julie A Shah. Collaborative planning with encoding of users' high-level strategies. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Shivam Miglani and Neil Yorke-Smith. Nltopddl: One-shot learning of pddl models from natural language process manuals. In *ICAPS'20 Workshop on Knowledge Engineering for Planning and Scheduling (KEPS'20). ICAPS*, 2020.

Roma Patel, Roma Pavlick, and Stefanie Tellex. Learning to ground language to temporal logical form. In *NAACL*, 2019.

David Abel, Will Dabney, Anna Harutyunyan, Mark K Ho, Michael Littman, Doina Precup, and Satinder Singh. On the expressivity of markov reward. *Advances in Neural Information Processing Systems*, 34:7799–7812, 2021.

Pashootan Vaezipoor, Andrew C Li, Rodrigo A Toro Icarte, and Sheila A Mcilraith. Ltl2action: Generalizing ltl instructions for multi-task rl. In *International Conference on Machine Learning*, pages 10497–10508. PMLR, 2021.

Srinivasan Iyer, Alvin Cheung, and Luke Zettlemoyer. Learning programmatic idioms for scalable semantic parsing. *arXiv preprint arXiv:1904.09086*, 2019.

Dan Goldwasser and Dan Roth. Learning from natural instructions. *Machine learning*, 94(2):205–232, 2014.

Ekin Akyürek and Jacob Andreas. Compositionality as lexical symmetry. *arXiv preprint arXiv:2201.12926*, 2022.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *arXiv preprint arXiv:2106.01345*, 2021.

Saurav Sahay, Eda Okur, Shachi H Kumar, and Lama Nachman. Low rank fusion based transformers for multimodal sequences. *arXiv preprint arXiv:2007.02038*, 2020.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Charles Packer, Pieter Abbeel, and Joseph E Gonzalez. Hindsight task relabelling: Experience replay for sparse reward meta-rl. *Advances in Neural Information Processing Systems*, 34:2466–2477, 2021.

Cheng Xue, Vimukthini Pinto, Chathura Gamage, Ekaterina Nikonova, Peng Zhang, and Jochen Renz. Phy-q: A testbed for physical reasoning, 2022.

Marlos C Machado, Marc G Bellemare, Erik Talvitie, Joel Veness, Matthew Hausknecht, and Michael Bowling. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Sam Shleifer, Jason Weston, and Myle Ott. Normformer: Improved transformer pretraining with extra normalization. *arXiv preprint arXiv:2110.09456*, 2021.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. *Advances in Neural Information Processing Systems*, 32, 2019.

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. *arXiv preprint arXiv:2206.14244*, 2022.

Jingkang Wang, Yang Liu, and Bo Li. Reinforcement learning with perturbed rewards. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 6202–6209, 2020.