Bootstrapped Q-learning with Context Relevant Observation Pruning to Generalize in Text-based Games

Subhajit Chaudhury
IBM Research
subhajit@jp.ibm.com

Daiki Kimura
IBM Research
daiki@jp.ibm.com

Kartik Talamadupula
IBM Research
krtalamad@us.ibm.com

Michiaki Tatsubori
IBM Research
mich@jp.ibm.com

Asim Munawar
IBM Research
asim@jp.ibm.com

Ryuki Tachibana
IBM Research
ryuki@jp.ibm.com

Abstract

We show that Reinforcement Learning (RL) methods for solving Text-Based Games (TBGs) often fail to generalize on unseen games, especially in small data regimes. To address this issue, we propose **C**ontext **R**elevant **E**pisodic **S**tate **T**runcation (**CREST**) for irrelevant token removal in observation text for improved generalization. Our method first trains a base model using Q-learning, which typically overfits the training games. The base model's action token distribution is used to perform observation pruning that removes irrelevant tokens. A second bootstrapped model is then retrained on the pruned observation text. Our bootstrapped agent shows improved generalization in solving unseen TextWorld games, using 10x-20x fewer training games compared to previous state-of-the-art (SOTA) methods despite requiring fewer number of training episodes.[1]

1 Introduction

Reinforcement Learning (RL) methods are increasingly being used for solving sequential decision-making problems from natural language inputs, like text-based games (Narasimhan et al., 2015; He et al., 2016; Yuan et al., 2018; Zahavy et al., 2018) chat-bots (Serban et al., 2017) and personal conversation assistants (Dhingra et al., 2017; Li et al., 2017; Wu et al., 2016). In this work, we focus on Text-Based Games (TBGs), which require solving goals like *"Obtain coin from the kitchen"*, based on a natural language description of the agent's observation of the environment. To interact with the environment, the agent issues text-based action commands ("*go west*") upon which it receives a reward signal used for training the RL agent. TBGs serve as testbeds for interactive real-world tasks like virtual-navigation agents on cellular phones at a shopping mall with user rating as the reward.

Goal: Who's got a virtual machine and is about to play through an fast paced round of textworld? You do! Retrieve the coin in the balmy kitchen.

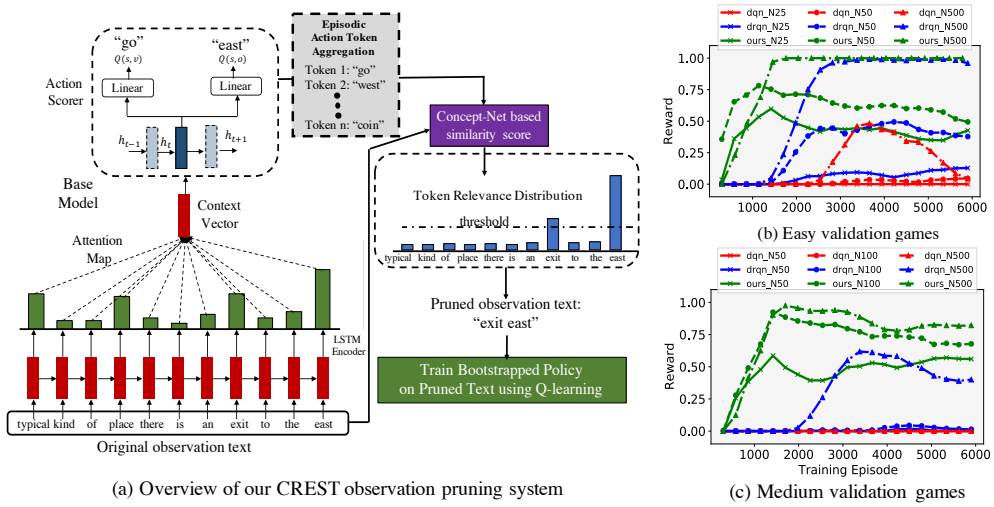
Observation: You've entered a studio. **You try to gain information on your surroundings by using a technique you call "looking."** You need an unguarded exit ? you should try going **east**. You need an unguarded exit? You should try **going south**. **You don't like doors?** Why not try going **west**, that entranceway is unblocked.
Bootstrapped Policy Action: *go south*

Figure 1: Our method retains context-relevant tokens from the observation text (shown in green) while pruning irrelevant tokens (shown in red). A second policy network re-trained on the pruned observations generalizes better by avoiding overfitting to unwanted tokens.

Traditional text-based RL methods focus on the problems of partial observability and large action spaces. However, the topic of generalization to unseen TBGs is less explored in the literature. We show that previous RL methods for TBGs often show poor generalization to unseen test games. We hypothesize that such overfitting is caused due to the presence of irrelevant tokens in the observation text, which might lead to action memorization. To alleviate this problem, we propose **CREST**, which first trains an overfitted *base* model on the original observation text in training games using Q-learning. Subsequently, we apply observation pruning for each training game, such that observation tokens that are not semantically related to the base policy's action tokens are pruned. Finally, we re-train a bootstrapped policy on the pruned observation text using Q-learning that improves generalization by removing irrelevant tokens. Figure 1 shows an

[1] Our code is available at: www.github.com/IBM/context-relevant-pruning-textrl

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 3002–3008,
November 16–20, 2020.



(a) Overview of our CREST observation pruning system

(b) Easy validation games

(c) Medium validation games

Figure 2: (a) Overview of Context Relevant Episodic State Truncation (CREST) module using Token Relevance Distribution for observation pruning. Our method shows better generalization from 10x-20x less number of training games and faster learning with fewer episodes on (b) "easy" and (c) "medium" validation games.

illustrative example of our method. Experimental results on TextWorld games (Côté et al., 2018) show that our proposed method generalizes to unseen games using almost 10x-20x fewer training games compared to SOTA methods; and features significantly faster learning.

2 Related Work

LSTM-DQN (Narasimhan et al., 2015) is the first work on text-based RL combining natural language representation learning and deep Q-learning. LSTM-DRQN (Yuan et al., 2018) is the state-of-the-art on TextWorld CoinCollector games, and addresses the issue of partial observability by using memory units in the action scorer. Fulda et al. (2017) proposed a method for affordance extraction via word embeddings trained on a Wikipedia corpus. AE-DQN (Action-Elimination DQN) – a combination of a Deep RL algorithm with an action eliminating network for sub-optimal actions – was proposed by Zahavy et al. (Zahavy et al., 2018). Recent methods (Adolphs and Hofmann, 2019; Ammanabrolu and Riedl, 2018; Ammanabrolu and Hausknecht, 2020; Yin and May, 2019; Adhikari et al., 2020) use various heuristics to learn better state representations for efficiently solving complex TBGs.

3 Our Method

3.1 Base model

We consider the standard sequential decision-making setting: a finite horizon Partially Observable Markov Decision Process (POMDP), represented as (s, a, r, s'), where s is the current state, s' the next state, a the current action, and $r(s, a)$ is the reward function. The agent receives state description $\boldsymbol{s}_t$ that is a combination of text describing the agent's observation and the goal statement. The action consists of a combination of verb and object output, such as "go north", "take coin", etc. The overall model has two modules: a representation generator, and an action scorer as shown in Figure 2. The observation tokens are fed to the embedding layer, which produces a sequence of vectors $\boldsymbol{x}^t = \{x_1^t, x_2^t, ..., x_{N_t}^t\}$, where N_t is the number of tokens in the observation text for time-step t. We obtain hidden representations of the input embedding vectors using an LSTM model as $h_i^t = f(x_i^t, h_{i-1}^t)$. We compute a context vector (Bahdanau et al., 2014) using attention on the j^{th} input token as,

$$e_j^t = v^T \tanh(W_h h_j^t + b_{attn}) \quad (1)$$

$$\alpha_j^t = \text{softmax}(e_j^t) \quad (2)$$

where W_h, v and b_{attn} are learnable parameters. The context vector at time-step t is computed as the weighted sum of embedding vectors as $c^t = \sum_{j=1}^{N_t} \alpha_j^t h_j^t$. The context vector is fed into the action scorer, where two multi-layer perceptrons (MLPs), $Q(s, v)$ and $Q(s, o)$ produce the Q-values over available verbs and objects from a shared MLP's output. The original works of Narasimhan et al. (2015); Yuan et al. (2018) do

Table 1: Average success rate of various methods on 20 unseen test games. Experiments were repeated on three random seeds. Our method trained on almost 20× fewer data has a similar success rate to state-of-the-art methods.

Methods	Easy			Medium			Hard	
	N25	N50	N500	N50	N100	N500	N50	N100
LSTM-DQN (no att)	0.0	0.03	0.33	0.0	0.0	0.0	0.0	0.0
LSTM-DRQN (no att)	0.17	0.53	0.87	0.02	0.0	0.25	0.0	0.0
LSTM-DQN (+attn)	0.0	0.03	0.58	0.0	0.0	0.0	0.0	0.0
LSTM-DRQN (+attn)	0.32	0.47	0.87	0.02	0.06	0.82	0.02	0.08
LSTM-DRQN (+attn+dropout)	0.58	0.80	1.0	0.02	0.37	0.85	0.0	0.33
Ours (ConceptNet+no att)	0.47	0.5	0.98	**0.75**	0.67	**0.97**	0.62	**0.92**
Ours (Word2vec+att)	0.67	0.82	**1.0**	0.57	0.92	0.95	0.77	**0.92**
Ours (Glove+att)	0.70	**0.97**	**1.0**	0.67	0.72	0.90	0.1	0.63
Ours (ConceptNet+att)	**0.82**	0.93	**1.0**	0.67	**0.95**	**0.97**	**0.93**	0.88

not use the attention layer. LSTM-DRQN replaces the shared MLP with an LSTM layer so that the model remembers previous states, thus addressing the partial observability in these environments.

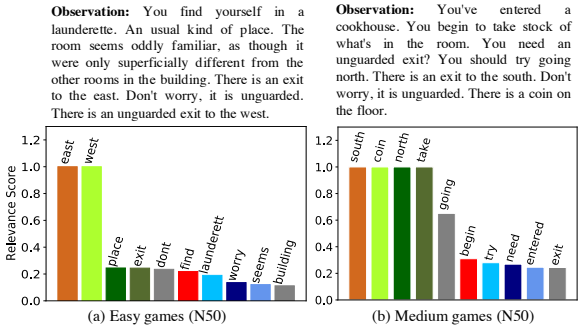
Q-learning (Watkins and Dayan, 1992; Mnih et al., 2015) is used to train the agent. The parameters of the model are updated by optimizing the following loss function obtained from the Bellman equation (Sutton et al., 1998),

$$\mathcal{L} = \left\| Q(s,a) - \mathbb{E}_{s,a}\left[r + \gamma \max_{a'} Q(s',a') \right] \right\|_2 \quad (3)$$

where $Q(s,a)$ is obtained as the average of verb and object Q-values, $\gamma \in (0,1)$ is the discount factor. The agent is given a reward of 1 from the environment on completing the objective. We also use episodic discovery bonus (Yuan et al., 2018) as a reward during training that introduces curiosity (Pathak et al., 2017) encouraging the agent to uncover unseen states for accelerated convergence.

3.2 Context Relevant Episodic State Truncation (CREST)

Traditional LSTM-DQN and LSTM-DRQN methods are trained on observation text containing irrelevant textual artifacts (like "You don't like doors?" in Figure 1), that leads to overfitting in small data regimes. Our **CREST** module removes unwanted tokens in the observation that do not contribute to decision making. Since the base policy overfits on the training games, the action commands issued by it can successfully solve the training games, thus yielding correct (observation text, action command) pairs for each step in the training games. Therefore, by only retaining tokens in the observation text that are contextually similar to the base model's



(a) Easy games (N50) (b) Medium games (N50)

Figure 3: Ranking of context-relevant tokens from observation text by our token relevance distribution.

action command, we can remove unwanted tokens in the observation, which might otherwise cause overfitting. Figure 2(a) shows an overview of our method.

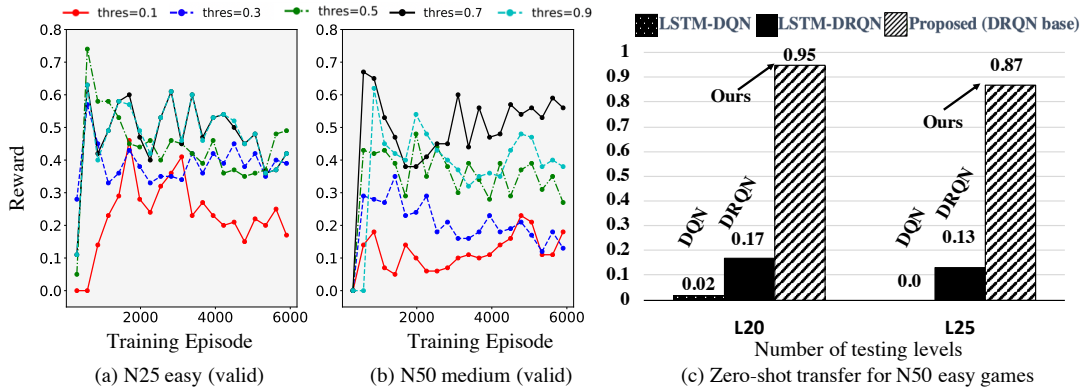
We use three embeddings to obtain token relevance: (1) Word2Vec (Mikolov et al., 2013); (2) Glove (Pennington et al., 2014); and (3) Conceptnet (Liu and Singh, 2004).

The distance between tokens is computed using cosine similarity, $D(\boldsymbol{a}, \boldsymbol{b})$.

Token Relevance Distribution (TRD): We run inference on the overfitted base model for each training game (indexed by k) and aggregate all the action tokens issued for that particular game as the Episodic Action Token Aggregation (EATA), $\mathcal{A}^k$. For each token w_i in a given observation text $\boldsymbol{o}_t^k$ at step t for the k^{th} game, we compute the Token Relevance Distribution (TRD), $\mathcal{C}$ as:

$$\mathcal{C}(w_i, \mathcal{A}^k) = \max_{a_j \in \mathcal{A}^k} D(w_i, a_j) \;\; \forall \, w_i \in \boldsymbol{o}_t^k, \quad (4)$$

where the i^{th} token w_i's score is computed as the maximum similarity to all tokens in $\mathcal{A}^k$. This relevance score is used to prune irrelevant tokens in the



(a) N25 easy (valid) (b) N50 medium (valid) (c) Zero-shot transfer for N50 easy games

Figure 4: Comparison of validation performance for various thresholds on (a) *easy* and (b) *medium* games, (c) Our method trained on $L15$ games and tested on $L20$ and $L25$ games significantly outperforms the previous methods.

observation text by creating a hard attention mask using a threshold value. Figure 3 presents examples of TRD's from observations highlighting which tokens are relevant for the next action. Examples of token relevance are shown in the appendix.

Bootstrapped model: The bootstrapped model is trained on the pruned observation text by removing irrelevant tokens using TRDs. The same model architecture and training methods as the base model are used. During testing, TRDs on unseen games are computed as $\mathcal{C}(w_i, \mathcal{G})$, by global aggregation of action tokens, $\mathcal{G} = \bigcup_k \mathcal{A}^k$, that combines the EATA for all training games. This approach retains all relevant action tokens to obtain the training domain information during inference, assuming similar domain distribution between training and test games.

4 Experimental Results

Setup: We used *easy*, *medium*, and *hard* modes of the Coin-collector Textworld (Côté et al., 2018; Yuan et al., 2018) framework for evaluating our model's generalization ability. The agent has to collect a *coin* that is located in a particular room. We trained each method on various numbers of training games (denoted by N#) to evaluate generalization ability from a few numbers of games.

Quantitative comparison: We compare the performance of our proposed model with LSTM-DQN (Narasimhan et al., 2015) and LSTM-DRQN (Yuan et al., 2018).

Figure 2(b) and 2(c) show the reward of various trained models, with increasing training episodes on *easy* and *medium* games. Our method shows improved out-of-sample generalization on validation games with about 10x-20x fewer training games (500 vs. $25, 50$) with accelerated training using drastically fewer training episodes compared to previous methods.

We report performance on unseen test games in Table 1. Parameters corresponding to the best validation score are used. Our method trained with $N25$ and $N50$ games for *easy* and *medium* levels respectively achieves performance similar to 500 games for SOTA methods. We perform an ablation study with and without attention in the policy network and show that the attention mechanism alone does not substantially improve generalization. We also compare the performance of various word embeddings for TRD computation and find that ConceptNet gives the best generalization performance.

Dropout: In Table 1, we also compare the performance of dropout (with probability 0.5) that randomly masks activations from the encoded state representation. We find that dropout improves performance compared to vanilla LSTM-DRQN. However, our method outperforms the model with dropout because dropout randomly drops tokens in an uninformed fashion. Our method uses a prior action token distribution from overfitted games to effectively remove irrelevant tokens.

Pruning threshold: In this experiment, we test our method's response to changing threshold values for observation pruning. Figure 4(a) and Figure 4(b) reveals that thresholds of 0.5 for easy games and 0.7 for medium games give the best validation performance. A very high threshold might remove relevant tokens, leading to failure in training, whereas a low threshold value would retain the most irrelevant tokens, leading to over-fitting.

Zero-shot transfer: In this experiment, agents

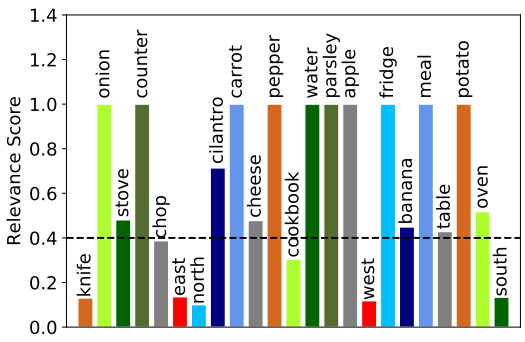


Figure 5: Token relevance scores for nouns in the test set for cooking games. The tokens having a score close to 1.0 correspond to overlaps between the train and test games. The other tokens were unseen during training. Our method can retain most tokens related to cooking using a threshold of 0.4, based on the training action token distribution obtained from an oracle.

trained on games with quest lengths of 15 rooms were tested on unseen game configurations with quest lengths of 20 and 25 rooms, respectively, without retraining, to study the zero-shot transferability of our learned agents to unseen configurations. The results in the bar charts of Figure 4(c) for $N50$ easy games show that our proposed method can generalize to unseen game configurations significantly better than previous state-of-the-art methods on the coin-collector game.

Generalizability to other games: In the above experimental section, we reported results on the coin-collector environment, where the nouns and verbs used in the train and test games have substantial overlap. We now present a discussion on our method's generalizability to other games, where the context-relevant tokens for a given game may never have occurred in any training game.

To test our method's generalizability, we performed experiments on the cooking games considered in Adolphs and Hofmann (2019). A sample observation from these games looks like this: "*You see a fridge. The fridge contains some water, a diced cilantro and a diced parsley. You wonder idly who left that here. Were you looking for an oven? Because look over there, it's an oven. Were you looking for a table? Because look over there, it's a table. The table is massive. On the table you make out a cookbook and a knife. You see a counter. However, the counter, like an empty counter, has nothing on it.*" The objective of this game is to prepare a meal by following the recipe found in the kitchen, and then eat it.

We took 20 train and 20 test games from the cooking domain, all featuring unseen items in the test observations. Training action commands were obtained from the oracle walkthrough games provided as part of the cooking world games, and not from the overfitted train games (since in this experiment we were evaluating the generalizability of the method across unseen tokens). From the training games, we obtain noun action tokens: {"onion", "potato", "parsley", "apple", "counter", "pepper", "meal", "water", "fridge", "carrot"}. Using our token relevance (TRD) method (using ConceptNet embeddings) described in Section 3.2, we obtain scores for unseen cooking related nouns during test as: {"banana": 0.45, "cheese": 0.48, "chop": 0.39, "cilantro": 0.71, "cookbook": 0.30, "knife": 0.13, "oven": 0.52, "stove": 0.48, "table": 0.43}.

Although these nouns were absent in the training action distribution, our proposed method can assign a high score to all these words (except "knife"), since they are similar in concept to the training actions. An appropriate threshold (for eg. th=0.4) can retain most tokens, as shown in Figure 5. The threshold value can be automatically tuned using validation games, as discussed in Section 4. Additionally, we believe that sampling action tokens from overfitted training games (our proposed method) instead of from an oracle (used for this result) would improve action token diversity and successfully retain more context-relevant words. Thus, assuming some overlap between training and testing knowledge domains, our method is generalizable and can reduce overfitting for RL in NLP tasks.

5 Conclusion

We present a method for improving generalization in TBGs by removing irrelevant tokens from observation texts. Our bootstrapped model – trained on the salient observation tokens – obtains generalization performance similar to SOTA methods – with 10x-20x fewer training games – due to better generalization; and shows accelerated convergence. In this paper, we have restricted our analysis to TBGs that feature similar domain distributions in train and test games. In the future, we will focus our attention on the topic of generalization in the presence of domain differences such as novel objects; and given goal statements in test games that were not seen by the agent during training.

References

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

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

Prithviraj Ammanabrolu and Matthew Hausknecht. 2020. Graph constrained reinforcement learning for natural language action spaces. *arXiv preprint arXiv:2001.08837*.

Prithviraj Ammanabrolu and Mark O Riedl. 2018. Playing text-adventure games with graph-based deep reinforcement learning. *arXiv preprint arXiv:1812.01628*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. 2018. Textworld: A learning environment for text-based games. *arXiv preprint arXiv:1806.11532*.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–495.

Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. 2017. What can you do with a rock? affordance extraction via word embeddings. *arXiv preprint arXiv:1703.03429*.

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

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. 2017. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 733–743.

Hugo Liu and Push Singh. 2004. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17.

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

Iulian V Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, et al. 2017. A deep reinforcement learning chatbot. *arXiv preprint arXiv:1709.02349*.

Richard S Sutton, Andrew G Barto, et al. 1998. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge.

Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning*, 8(3-4):279–292.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Xusen Yin and Jonathan May. 2019. Learn how to cook a new recipe in a new house: Using map familiarization, curriculum learning, and bandit feedback to learn families of text-based adventure games. *arXiv preprint arXiv:1908.04777*.

Xingdi Yuan, Marc-Alexandre Côté, Alessandro Sordoni, Romain Laroche, Remi Tachet des Combes, Matthew Hausknecht, and Adam Trischler. 2018. Counting to explore and generalize in text-based games. *arXiv preprint arXiv:1806.11525*.

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