# Deep Reinforcement Learning using Symbolic Representation for Performing Spoken Language Instructions*

Mohammad Ali Zamani, Sven Magg, Cornelius Weber, and Stefan Wermter

*Abstract*— **Spoken language is one of the most efficient ways to instruct robots about performing domestic tasks. However, the state of the environment has to be considered to plan and execute the actions successfully. We propose a system which can learn to recognise the user's intention and map it to a goal for a reinforcement learning (RL) system. This system is then used to generate a sequence of actions toward this goal considering the state of the environment. The novelty is the use of symbolic representations for both input and output of a neural Deep Q-network which enables it to be used in a hybrid system. To show the effectiveness of our approach, the Tell Me Dave corpus is used to train the intention detection model and in a second step to train the RL module towards the detected objective, represented by a set of state predicates. We show that the system can successfully recognise command sequences from this corpus as well as train the deep-RL network with symbolic input. We further show that the performance can be significantly increased by exploiting the symbolic representation to generate intermediate rewards.**

## I. INTRODUCTION

In the future, robots are expected to work as companions with humans in various areas ranging from domestic to caregiving scenarios. Even with well-engineered robots, it would be unrealistic to transfer them directly from factories to home environments to perform complex tasks such as caregiving [1], [2]. One of the main reasons is safety [3]. Moreover, the robots also have to adapt to new environments to perform the given tasks. Using experts to program robots for every environment is expensive. Hence, we need adaptive learning algorithms that can be "programmed" by the users.

Spoken language can be considered as one of the most effective communication channels to instruct robots to perform a sequence of actions to fulfill a task. Assigning tasks to robots by verbal instructions has been studied for example in [4], [5] and [6] but the problem had to be limited to small domains due to the variability in language and the corresponding problem to understand the human's intention.

Intention detection in spoken language has been studied to some extent on short texts [7] with a focus on applications like web search. Another project was started in the Facebook DeepText project[1] inspired by [8]. In the DeepText project, all the sentences "I need a ride", "Take a cab" "But, I need to take a taxi" are interpreted as "Request a ride" which reflect the user intention in a unique phrase.

Deep neural networks have achieved a significant improvement in recent years especially in domains like machine translation [9] and can also be used for intention classification. The encoder-decoder architecture enables training different lengths of sequences. One of the most successful implementations [10] used Long Short-Term Memory (LSTM) for both encoder and decoder [11].

Such an end-to-end approach for translation could be applied to map the spoken utterance (sequence of words) into a sequence of actions. The input and output consist of the sequences of text and actions, respectively. The proposed approach should be able to map the sentence "Put the mug into the microwave" into a sequence of low-level robot instructions like "*Moveto Mug, Grasp Mug, Moveto Microwave, Open Microwave, Put Mug in Microwave*". Although it seems similar to a machine translation task, due to the high variability of both language and actions that lead to fulfilment of the objective, the amount of training data and training time needed makes this approach currently not feasible. It is made even more complex since the current state of the environment has to be taken into account as well.

Reinforcement learning has to be proven to be an effective method to learn a task through interaction with the environment [12] and has been used in human-robot interaction (HRI) scenarios. In the RL framework, an agent interacts with the environment and receives rewards. The overall goal of the agent is to maximize the expected return (which is a kind of accumulation of rewards in each trial), often to reach a terminal state. To improve RL in an HRI context, [13] introduced a feedback message from the human that can be used in the interactive reinforcement learning (IRL) framework, either as guidance for action selection or as reward for evaluating the selected action. Thomaz and Breazeal (2008) implemented IRL in a simulated environment called "Sophie's Kitchen" in which the robot agent learned to bake a cake. The simulation allows the human to click on the object to guide the robot and also give a reward through clicking on a sliding bar [14]. However, the scenario was only designed for one fixed intention to limit the state space and thus the size of the Q-table.

Using deep convolutional neural networks as value function approximatiors [15] boosts RL to gain more powerful generalization capabilities for larger state spaces. Mnih et al. (2015) implemented deep reinforcement learning that could surpass human performance in many Atari-2600 games by reading only raw pixels from the screen. Deep Q-networks

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 642667 (SECURE).

Knowledge Technology, Department of Informatics, Universität Hamburg, Vogt-Kölln-Str. 30, 22527 Hamburg, Germany (www.knowledge-technology.info) {zamani, magg, weber, wermter} @informatik.uni-hamburg.de

[1]https://code.facebook.com/posts/181565595577955/introducing-deeptextfacebook-s-text-understanding-engine/

have also been implemented for text-based games [16]. The agent in the game environment receives a description of the situation in a few sentences which are encoded by an LSTM to represent the change in the game state. Then, a multi-layered neural network is used to score possible actions in the given state. Although the agent learns to perform the given quest, it does not use the environment's state representation directly but builds a representation of the instruction instead.

A symbolic representation for the environment's state on a more abstract level can help to solve the task. This representation can be obtained from the vision modality. For instance, [17] trained a deep convolutional neural network to represent a symbolic description of the given image. Garnelo et al. [18] implemented a system which extracts a symbolic representation from the environment. However, their action-value function used for RL was learned through discrete Q-learning which is not extendable to complex problems.

A symbolic representation enables hybrid systems that also include classic planning and deduction systems, and on the other hand, helps us to easily generate intermediate rewards for the agent because of compositionality in the state representation. The objective of our research is to develop a system that understands the intention of an utterance and uses it to generate a sensible sequence of actions. Therefore, we separated the problem into two steps, intention detection and action planning. We propose a novel deep reinforcement learning architecture that directly learns from the symbolic representation. In the results section, we show the feasibility of the approach for a task in a kitchen scenario.

## II. Corpora for Instructing Robots

There are only a few annotated corpora available for instructing robots which are large enough to be used for supervised learning. A corpus for natural language instruction to a robot is the "Tell Me Dave" corpus [19] and [20]. This corpus has 20 particular environments including 10 different kitchens and 10 different living rooms (with a complete symbolic description of the objects' position, orientation and their states). In the VEIL-500 (Verb-Environment-Instruction-Library) version of the corpus, the human users ask the robot to perform specific tasks, i.e. it provides sentences for 10 objectives (boiling water, cleaning the kitchen, etc.). Another group of subjects was asked to annotate the sequence of actions to successfully perform those tasks. The "Tell Me Dave" corpus is the largest language corpus annotated for action planning to the best of our knowledge. The environments in the corpus are designed similar to a home environment and are thus suitable for our work and future extensions.

## III. Method

We propose a modular approach which is divided into sub-systems (figure 1) with measurable sub-goals including intention detection and reinforcement learning (action planning). In the case of significant change of the environment, it is thus not necessary to retrain all modules, but only the reinforcement learning has to be updated. Unlike supervised learning, the RL module does not need any corresponding pair of instructions and actions. Instead, it learns through interaction with the environment, towards a goal and receives sparse feedback which consists only of a positive scalar number for completing the task (or subtasks).

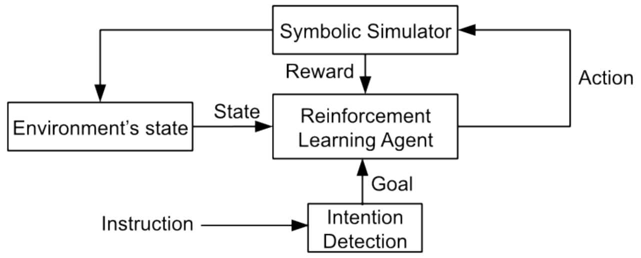

Fig. 1. The modular approach using intention detection and reinforcement learning trained for each objective to generate the sequence of actions.

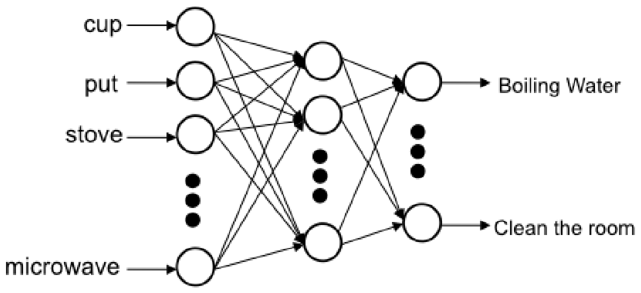

Fig. 2. An MLP neural network is used to implement the intention detection. The Tell Me Dave corpus is used to train the neural network.

In the current format different RL modules can be trained for each intention (e.g. boil water, clean the room) to avoid the complexity of having multiple objectives in the RL for now. Therefore, a classifier detects the intention of the given instruction and triggers the associated RL module to generate the sequence of actions.

### A. Intention Detection

The Tell Me Dave corpus already provided the objectives of the tasks, which we can use to extract a goal for reinforcement learning. The total vocabulary size in the Tell Me Dave corpus is 687. A given instruction was represented as a binary vector (same size of the vocabulary) which indicated whether each word was present in the given instruction. A Multi-Layer Perceptron (MLP) with one hidden layer was used to classify among 10 existing objectives in the corpus. In the proposed architecture, the hidden layer had 50 nodes with ReLU as activation function as well as 20% dropout. The output layer included 10 nodes equal to our classes with softmax as the activation function. As loss function, Categorical Cross-entropy minimized by an Adam optimizer was used as introduced in [21].

### B. Symbolic Representation

Given the objective, the corresponding RL module performs the action sequence. The RL agent receives the symbolic representation of the environment consisting of predicates which each describe either a spatial relation between two objects, e.g. *(On Mug Sink)*, or the state of an object, e.g. *(State Mug Water)*. Combinations of these predicates, regardless of their order, shape the state of the environment. This creates a huge amount of possible combinations of the predicates in the state representation which would make a Q-table an interactable model. The number of the predicates describing the current state is also not constant, not only for different environments but also during the performance of a specific task. Moreover, we are seeking to see generalization in the presence of some predicates with combinations that are seen by the agent for the first time.

The symbolic information of the environment is given as $\chi(t) = \{x_1, x_2, ..., x_k\}$, where $\chi$ is the set of predicates which represent the state of the environment, $t$ is the time step, $k$ is the maximum number of existing predicates in each time step. Each predicate $x_i$ is a member of all possible predicates $\Pi = \{x_1,\ x_2,\ ...,\ x_d\}$, where $d$ is total number of possible predicates which in our case is 180. We convert the symbolic state $\chi(t)$ to a binary vector of fixed length

$$s(t) = [q_1,\ ...,\ q_d]\,,\ where \underset{i \in \{1,...,d\}}{q_i} = \begin{cases} 1, & if\ x_i\ \in\ \chi(t) \\ 0, & if\ x_i\ \notin\ \chi(t) \end{cases}$$

We developed a symbolic simulator based on the domain knowledge (i.e. Planning Domain Definition Language (PDDL) file) in the Tell Me Dave corpus to train the RL agent. PDDL is the standard to represent an action planning task. It defines the effects as well as preconditions of each action on the environment. The symbolic simulator on the basis of this description was developed based on the OpenAI gym environment [22].

## C. Deep Reinforcement Learning

Similar to the approach in [16], outputs are grouped into actions and objects. However, we needed to add two more output groups for the second object and the preposition which is necessary for some actions (i.e. for the action *keep* in *(keep mug on sink)*). The output of the network is thus divided into four group of outputs as *action, object 1, object 2* and *preposition*. When a group of outputs is not valid, their nodes are masked within the learning to not affect the gradient. There is also no reason to use a convolutional neural network as in [9] or an LSTM as in [16] because the input of the deep reinforcement learning uses already an abstract state representation instead of raw image pixels or plain texts. Since the model has the Markov Decision Property (see [12]) the state is already adequate to fully represent the environment. Therefore, a simpler MLP was used to approximate the action-value (Q) function, which is presented in figure 3. Due to the complexity of the task, one hidden layer was not enough to yield a stable solution. Therefore, we added 4 hidden layers to gain enough capabilities within the network. The hidden layers' activation functions are ReLU and the output functions are linear. The parameters of the network were found empirically.

The loss function for the deep RL was defined as introduced in [15]

$$L_i(\theta_i) = E_{(s,a,r,s')}\left[r + \gamma \max_{a'} Q(s^{'}, a^{'}; \bar{\theta}_t) - Q(s, a; \theta_t)\right]^2$$

where $\gamma$ is the discount factor (0.9), and *r* is the reward that the agent receives from the environment.

The agent selected an action based on an epsilon-greedy policy ($\epsilon = 0.1$) and the observed experience was recorded as $e_t = \{s_t,\ a_t,\ r_t,\ s_{t+1}\}$ in the memory. The memory recorded the last 100,000 experiences. Meanwhile, in each time step, a batch (32 in our experiment) of experience was replayed to train the network ($\theta$) to avoid teaching highly correlated samples in a row, as introduced by [15]. Beside the experience replay, to gain more stability, a duplicate neural network which is called "target network" ($\bar{\theta}$) [15] was updated every 10,000 iterations from the trainable network (i.e. $\theta$).

*1) Reward:* We implemented two approaches for obtaining rewards from the environment. First, the agent only got reward when it reached the goal state (i.e. reward of 1). In the second approach, the agent received an intermediate reward (0.1) when it found a predicate present in the goal state for the first time in every episode[2]). If the goal state had $k$ predicates, the agent was able to receive a maximum of $k-1$ intermediate rewards (regardless of required steps) plus one terminal reward. For all other steps, the agent received no reward.

*2) Memory:* In our RL problem, an agent receives sparse rewards during the learning process. A random batch sampling from the memory is not efficient to train such an agent. Therefore, we applied rank-based prioritized experience replay (PER) as introduced by [23]. In this approach, TD samples are ranked based on the TD-error. Due to multiple outputs in our model, the TD-error was modified as

$$\delta_{e_t} = \frac{1}{V}\sum_{v=1}^{V}\left|r + \gamma \max_{a_v^{'}} Q(s^{'}, a_v^{'}; \bar{\theta}_t) - Q(s, a_v; \theta_t)\right|$$

where $\delta_{e_t}$ is the TD-error for the experience ($e_t$), and V the number of existing outputs for the experience (this is variable depending on the action. E.g. in *(grasp, cup)*, $V = 2$ *and in (keep, kettle, stove, on)*, $V = 4$). Similar to [23], the priority of the sample was $p(e_t) = \frac{1}{rank(e_t)}$, and the probability of selecting the sample thus $P(e_t) = \frac{e_t^{\alpha}}{\sum_k e_t^{\alpha}}$ where $\alpha$, the prioritization factor, was 0.7 as recommended by [23]. After each sampling, the probability of each experience was updated in the memory. For an efficient sampling, the memory was also implemented as a binary heap, which was rebalanced every 10 episodes. We also implemented the importance sampling weights recommended by Schaul *et al* [23] to correct the bias caused by the PER that violates the distribution of experiences as they are normally expected.

## IV. Results and Discussions

The intention detection network was first trained for the Tell Me Dave corpus. There were 469 valid samples in the

[2]Each episode had a maximum of 500 iterations in our implementation.

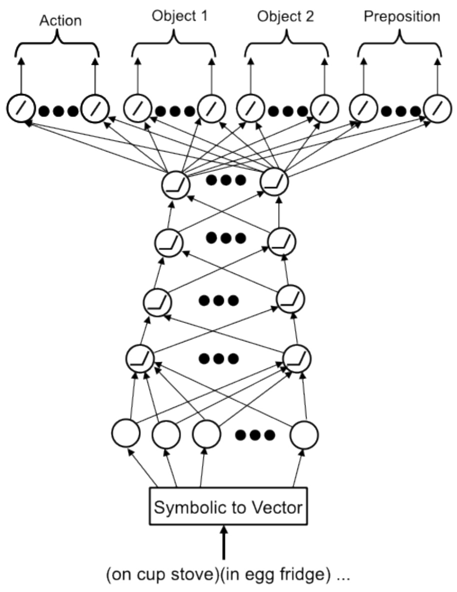

Fig. 3. The deep reinforcement learning architecture. An MLP neural network is trained to approximate the action-value functions. The "Symbolic to Vector" modules (see section III.B) converts the given predicates to a binary vector.

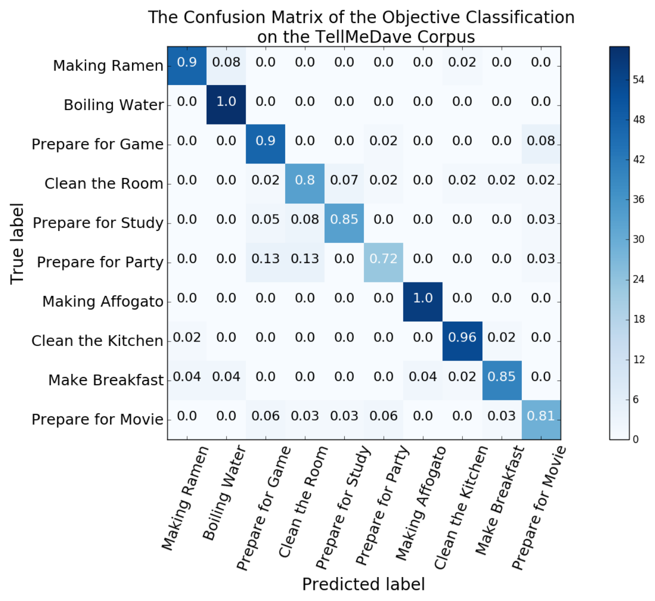

Fig. 4. Confusion matrix for the classification of objectives from the Tell Me Dave corpus.

VEIL-500 version of the corpus. The results for a 5-fold cross validation show (see table I) that the objectives of the instructions can be classified accurately on average in 89.57% of cases from the test set. An early stopping was used by monitoring the loss function on a validation set of the size 5% of the training set each time. A single layer perceptron (i.e. no hidden units) was selected as the baseline which could obtain 85.74% accuracy. The current intention detection mechanism could also be implemented with a non-neural method such as keyword spotting. However, the proposed architecture can more easily be extended to also classify not only objectives but also feedback and warning from a human in the future.

The confusion matrix (figure 4) shows that the most confusion happened between "Prepare for party ", "Preparing for the study night" and "Clean the room" which have more common words. For example, two similar sample instructions from these two classes are: "Take all bottles, cans, and trash and put them in the trash can ..." as the "Preparing for the study night" intention and "... Put beer and chips in trash can." as the "Clean the room" intention.

The deep RL performance was first evaluated for the two approaches with and without using the intermediate reward for the *boiling water* objective. The goal state for the training was derived by executing the annotated minimal action sequence in the simulator. The RL network thus had to learn the action sequence that leads to this goal state. The results are presented in (table II) for 9 randomly initialized trials for each approach. It can be seen that the agent which was trained with the intermediate reward method, was almost twice as fast for this objective than the one that learned with only the terminal reward signal (t-test p-value = 0.0014). This shows that using the intermediate rewards improves the speed of learning for this scenario despite ignoring the necessary order of goal predicates in the reward process, e.g. putting the kettle on the stove without water leads to the intermediate reward. This means, no knowledge about the order of sub-tasks is used.

We also investigated the performance gains of using the prioritized experience replay method by comparing it to a standard random sampling of experiences. As table II shows, RL with terminal reward signal benefits clearly by prioritizing experiences. Interestingly, if the intermediate rewards are given, then the PER shows no significant performance gains.

## V. CONCLUSIONS AND FUTURE WORK

Our approach uses two steps to combine deep reinforcement learning with symbolic representations: First detect the

TABLE I
OBJECTIVE CLASSIFICATION RESULTS. THE NUMBERS SHOW THE MEAN AND STANDARD DEVIATION PERCENTAGE.

| | Training Acc. | Validation Acc. | Test Acc. |
|---|---|---|---|
| MLP | $99.73 \pm 0.18$ | $99.37 \pm 1.39$ | $89.57 \pm 4.27$ |
| Perceptron | $97.44 \pm 0.07$ | $96.87 \pm 4.42$ | $85.74 \pm 3.91$ |

TABLE II
ITERATIONS UNTIL THE AGENT LEARNED TO PERFORM THE TASK FOR THE FIRST TIME WITH FULL EXPLOITATION.

| | rank-based PER | Random Sampling |
|---|---|---|
| only terminal reward | $201877 \pm 47741$ | $452745 \pm 362745$ |
| intermediate reward | $103190 \pm 60313$ | $115570 \pm 63847$ |

objective of the given linguistic instruction and secondly generate the sequence of actions based on the symbolic representation of that objective. In this paper, we presented a proof-of-concept to show the general utility of this hybrid approach. Furthermore, we show that using a symbolic representation, intermediate rewards can easily be determined, which boosts the RL learning performance.

The current implementation of the intention classification is only based on spotting keywords in the given sequence to obtain the objective. The performance could be improved if the input is represented using word embeddings [24] and using recurrent neural networks [25] because the instructions are given as word sequence. We also intend to not only classify objectives in the future, but to extend the classifier to distinguish user feedback such as warnings and to incorporate this in the learning process (Interactive RL). With this extension, it is then possible to replace the current goal state extraction through simulation with a learning system that learns which predicates are necessary to satisfy the user's intention.

By splitting up the intention detection and the goal-directed learning, the presented RL framework only has to rely on the signal from the environment to understand whether the agent is in the terminal state. By using symbolic representations at both ends of the Deep-Q Network, the input and output can be interpreted easily and the network also be used within a hybrid system. We showed by using one objective that such a system can be trained towards a simple task in a home scenario effectively. Several objectives can be used by training different networks and selecting the correct one for the given task after intention detection. The RL network also offers the complexity to be trained towards several goals and thus to combine several objectives within the same network. By doing this, common sequences to reach sub-goals can be efficiently exploited.

Our main finding was the performance gains that can be obtained by using the individual predicates of the symbolic goal state as intermediate rewards in the learning process. These intermediate rewards can be easily generated, as we have shown, by rewarding the appearance of individual predicates that need to be satisfied once when they are encountered first. Since no domain knowledge was necessary to order those predicates, our method provides a simple way to speed up learning by using symbolic representations. Compared to other symbolic approaches, our model does not need any explicit internal representation of the world model. Moreover, the deep neural network, which approximates the action value function, generalizes over the states to select an action. We thus showed that combining symbolic representations with neural reinforcement learning not only makes it possible to implement a hybrid system but it also improves the performance by generating intermediate rewards, which are readily available in this representation.

## References

[1] S. Schaal, "The new robotics–towards human-centered machines," *HFSP journal*, vol. 1, no. 2, pp. 115–126, 2007.

[2] S. Schaal and C. G. Atkeson, "Learning control in robotics," *IEEE Robotics & Automation Magazine*, vol. 17, no. 2, pp. 20–29, 2010.

[3] J. Peters and S. Schaal, "Learning to control in operational space," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 197–212, 2008.

[4] S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein, "Mobile robot programming using natural language," *Robotics and Autonomous Systems*, vol. 38, no. 3, pp. 171–181, 2002.

[5] S. Lauria, G. Bugmann, T. Kyriacou, J. Bos, and E. Klein, "Converting natural language route instructions into robot executable procedures," in *Robot and Human Interactive Communication, 2002. Proceedings. 11th IEEE International Workshop on*. IEEE, 2002, pp. 223–228.

[6] T. Nishizawa, K. Kishita, Y. Takano, Y. Fujita, *et al.*, "Proposed system of unlocking potentially hazardous function of robot based on verbal communication," in *System Integration (SII), 2011 IEEE/SICE International Symposium on*. IEEE, 2011, pp. 1208–1213.

[7] W. Hua, Z. Wang, H. Wang, K. Zheng, and X. Zhou, "Short text understanding through lexical-semantic analysis," in *Data Engineering (ICDE), 2015 IEEE 31st International Conference on*. IEEE, 2015, pp. 495–506.

[8] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press Cambridge, 1998, vol. 1, no. 1.

[13] A. L. Thomaz, G. Hoffman, and C. Breazeal, "Real-time interactive reinforcement learning for robots," in *AAAI 2005 workshop on human comprehensible machine learning*, 2005.

[14] A. L. Thomaz and C. Breazeal, "Teachable robots: Understanding human teaching behavior to build more effective robot learners," *Artificial Intelligence*, vol. 172, no. 6-7, pp. 716–737, 2008.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[16] K. Narasimhan, T. Kulkarni, and R. Barzilay, "Language understanding for text-based games using deep reinforcement learning," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal, 2015, pp. 1–11.

[17] A. Kumar and T. Oates, "Connecting deep neural networks with symbolic knowledge," in *The 2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 3601–3608.

[18] M. Garnelo, K. Arulkumaran, and M. Shanahan, "Towards deep symbolic reinforcement learning," *arXiv preprint arXiv:1609.05518*, 2016.

[19] D. K. Misra, J. Sung, K. Lee, and A. Saxena, "Tell Me Dave: Context-sensitive grounding of natural language to manipulation instructions," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 281–300, 2016.

[20] D. K. Misra, K. Tao, P. Liang, and A. Saxena, "Environment-driven lexicon induction for high-level instructions." in *ACL (1)*, 2015, pp. 992–1002.

[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, San Diego, 2015.

[22] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv:1606.01540*, 2016.

[23] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[24] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543, 2014.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv*, pp. 1–9, 2014. [Online]. Available: http://arxiv.org/pdf/1412.3555v1.pdf