Interactive Learning with Natural Language Feedback

David J Fernandez [†1] , Guillermo N Grande[†1] , Sean C Ye[†1],
Nakul Gopalan[1] , and Matthew Gombolay[12]

***Abstract*—We seek to enable non-roboticist end-users to teach robots by examining how end-users can provide natural language feedback to the robot. We hypothesized that enabling users to use language to train an agent would be more intuitive as user's don't have to translate their intent through another system. We build upon Deep TAMER to allow users to provide feedback through natural language to a learning agent. Our algorithm includes (1) a Transformer based language model to map natural language feedback to scalar reward values and (2) a method to synthetically assign rewards to nearby state-action pairs that were unexplored by the agent. We report our results from a 2x4 mixed-subjects experiment design to evaluate the usability, workload, and trainablitity of our system compared to Deep TAMER on simulated tasks. While the experimenters were able to train an agent on both simulated environments to achieve competitive rewards, we could not show that natural language feedback significantly lowered workload, increase usability, or train better agents than baseline Deep TAMER with human subjects. This work indicates a need for further research into the types of feedback end-users prefer to use to train agents.**

***Index Terms*—Robot Learning, Language Feedback**

I. Introduction

Advancements in robot learning have created the opportunity for benefiting people across a wide variety of applications, including healthcare [1], manufacturing [2], and assistive household tasks [3]. However, deploying robots in real-world settings typically require experts to design and program these robots. This work explores how end-users without expert knowledge can provide feedback to learning robotic agents through language.

Prior work has investigated how robots can use human feedback as reward signal for learning novel skills. Two such works on training agents with human feedback are TAMER [4], [5] and COACH [6]. In these works, the human teacher provides the agent with real time feedback usually through the form of a clicker or keyboard.

We posit that natural language may provide a more intuitive interface for users to train robots and a more informative feedback signal for robots. In this work, we utilize the tremendous progress the Natural Language Processing (NLP) community has made in sentiment classification [7], [8] to convert natural language to a continuous feedback scale. Using natural language inputs from a teacher as a feedback signal provides the advantage that the feedback scale being used is natural and intuitive for the teacher. As such, we hypothesize that providing consistent feedback signals becomes an easier task, given that the teacher does not have to reason about translating their intent to an unfamiliar scale. We therefore leverage sentiment scores generated by extending DistilBERT [8] as continuous signals for providing human feedback to agents.

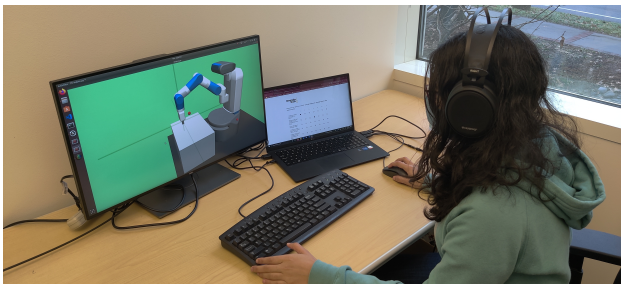

Fig. 1. **Experimental Setup:** Participant training the Fetch-Reach task with language feedback in our experimental setup

One problem associated with using natural language as feedback signals is the inevitable increase in feedback signal sparsity when compared to traditional feedback signaling (i.e. it is faster to provide a keyboard input than a verbal cue). We choose to use a data augmentation method similar to [9] to create Gaussian hyperspheres that assign feedback to more states. A key difference from our implementation is that our approach generalizes to tasks with arbitrary goal state definitions. We assign artificial states a feedback value given by a weighted average of the teacher provided feedback signal and the prediction of a target network as seen in [10].

Our goal was to extend the Deep TAMER algorithm to allow users to train agents with more expressive feedback. We hypothesized that training agents with natural language commands compared to keyboard input would reduce workload, increase usability, and result in trained agents that performed better than baseline agents. In this paper, we (1) develop a novel robot training algorithm that maps language to continuous feedback, (2) create a method to artificially assign feedback to nearby state-action pairs, and (3) conducted a user study analyzing our method compared to Deep TAMER on two Open-AI Gym domains: Fetch-Reach [11] and MountainCar [12].

II. Related Work

Using human feedback to train agents has been successful with discrete valued feedback [4], [6]. Deep learning has enabled training to also work in high-dimensional image state spaces [5]. In this work, we build upon these frameworks in to include natural language feedback.

The first three authors contributed equally to this work. † indicates corresponding authors.

[1]Georgia Institute of Technology, Atlanta, GA {dfernandez33, nicogrande, seancye, nakul}@gatech.edu

[2] School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA matthew.gombolay@cc.gatech.edu

A. Teaching with Natural Language

Using natural language feedback to help guide policy convergence in reinforcement learning is a problem increasingly being studied today. Krening et al. [13] leverage sentiment classification to classify human verbal feedback as either advice or warning in the Super Mario Bros. domain. The authors utilize an object-focused approach which parses the language with a predefined structure on environment objects. The authors then use these explanations to train a policy without state information. While we also leverage sentiment classification, we use the sentiment to directly extract a score representative of the relative benefit of a particular state-action pair determined by a human teacher. Additionally, we do not reduce natural language inputs with a predefined structure which makes our approach task agnostic.

Harrison et al. [14] improve on [13] by enabling task agnostic training of agents via natural language human feedback. They enable task agnostic training via natural language by eliminating the requirement of additional information about the environment, such as descriptions of object types, and instead opt to use an encoder-decoder network to learn associations between verbal feedback and state-action pairs. Harrison et al. then use the encoder-decoder network to guide the exploration of a reinforcement learning agent through policy shaping. Training this encoder-decoder network is done offline by the human demonstrator for a specific task, which is not required by our approach.

B. Training Agents with Feedback

Training agents with feedback has been successful in a variety of scenarios including Tetris [4] and learning physical robot behaviors [6]. Our approach also differs from [14] in that we build on Deep TAMER [5], learning directly from human feedback as opposed to policy shaping [15]. In Deep TAMER, the human provides feedback in real-time observing an autonomous agent perform a task and providing a scalar-valued feedback which shapes the agent behavior. Through this feedback, the agent learns the parameters of a deep neural network which predicts the human's feedback. The agent then uses the prediction of human feedback to drive behavior, essentially functioning as its policy. It is often difficult for a human demonstrator to discern the exact action taken by an agent in a given state while giving online advice. We therefore try to alleviate this effect by inferring the relative value of a particular state in our selected domains. A more recent method for natural language guided reinforcement learning is A3PS [16]. This method features a pretrained advice generator network which generates human-like feedback for a given state observation. This feedback is then used to generate advice conditioned action scores which are then combined with a PPO agent's action scores to generate a final action. A key difference between our work and A3PS is that A3PS does not take in real time human feedback. Furthermore, the A3PS advice generator must be pretrained using state-advice pairs which constrains their framework to specific tasks.

C. Data Augmentation

Data augmentation methods to aid reinforcement learning policy convergence are also crucial to our work, particularly due to the sparse nature of natural language human feedback. Lin et al. [9] present an approach similar to ours, in which for a given robotics domain episode, goal states are artificially generated at fixed reflections of the original goal. This technique enables the robot's predicted trajectory for the original goal to also be reflected, thereby generating artificial state-action pairs in high dimensional continuous state spaces. Since we cannot assume a symmetrical nature in goal states for the vast majority of tasks, we instead choose to generate artificial state-action states using a Gaussian hypersphere about each state encountered by our agent. The action selected at each artificial state generated is determined by a target network as specified in Section III.

III. Method

We build on Deep TAMER by introducing three new components: (a) A language model for sentiment classification, (b) HyperSphere noise injection, and (c) a target network to help convergence of our feedback signal.

a) Language Model: We built a Transformer based architecture to allow users to provide natural language feedback for training agents. Our model extends DistilBERT by replacing the final classification Softmax layer with a liner transformartion to yield a training signal. We train this layer with the Stanford Tree Bank Sentiment dataset [17] and map the ground truth labels of [very negative, negative, neutral, positive, very positive] to [-2, -1, 0, 1, 2].

b) HyperSphere Noise Injection: Given the formulation of feedback shaping via TAMER [4], we identified a key issue associated with providing sparse feedback signals in a continuous state space: *many state-action tuples will be unexplored by the agent, and hence, never be assigned an expert feedback signal.* To solve this issue, we provide the same feedback to unexplored state-action tuples near explored state-action tuples. We call this procedure "hypersphere noise-injection", which generates artificial state-action tuples for the agent and assigns them a ground truth feedback signal. The formulation of hypersphere noise-injection is formalized as follows. Given an N dimensional continuous state, $\mathbf{s} = s_0, s_1, \ldots, s_N$, where the value at each dimension s_i for $0 \leq i \leq N$ takes a single continuous scalar value within the bounds of the state space S, we define an N-dimensional multivariate Gaussian distribution $\mathcal{N} = (\mu, \Sigma)$, where μ and Σ are defined as follows:

$$\mu = \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_N \end{pmatrix} \quad \Sigma = \begin{pmatrix} c \cdot s_0 & 0 & \ldots & 0 \\ 0 & c \cdot s_1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & c \cdot s_N \end{pmatrix} \tag{1}$$

We then sampled k artificial states from $\mathcal{N} = (\mathbf{s}, \texttt{Diag}(c \cdot \mathbf{s}))$. The c value is a scaling factor to generate a standard deviation proportional to the magnitude of the corresponding mean. If the distribution is too large, the human feedback (h_t)

may be assigned to states to which the human did not intend and reduce how well the agent can learn. If the distribution is too small, the system may not generalize well enough to surrounding states and the problem remains of leaving many state-action pairs unlabeled. Thus a limitation is that the feedback given to artificial states may be different than what the human intended if nearby state-action pairs are not similar.

With this formulation, we force our agent to artificially receive feedback on more states in the continuous state space, thereby extracting more information with the same amount of sparse feedback signals from the human. We experimented with values of c and chose $c = 0.1$ but would like to conduct a formal analysis of the effect of varying this parameter in later work.

In the next section, we introduce our approach to partially account for the assumption that all artificial states $\mathbf{s}' \sim \mathcal{N} = (\mathbf{s}, \texttt{Diag}(c \cdot \mathbf{s}))$ can be assigned the same ground truth feedback value h_t given an observed action and provide artificial states with feedback values also dependant on the artificial states themselves.

c) Target Network: A flaw with the approach presented in Section III-b is the way in which these artificially generated states contribute to the overall loss of a given $(\mathbf{s}_t, a_t, h_t, c_t)$ tuple. Therefore, we relax the assumption that the feedback value of sampled states is equivalent to the original state by utilizing a target network, first proposed by Mnih et al [10].

A target network allows us to query our learned feedback estimator to obtain more accurate information about the feedback associated with the artificial states while preventing the negative effects of regressing onto ourselves, since these values are incorporated into our objective function defined in Equation 4.

Querying the learned feedback estimator is not necessarily a proper way of gauging the feedback associated with a state, specifically in the early stages of training. To account for this difference in performance at different points in time, we compute the feedback for artificially generated states as a weighted sum of the human feedback signal and the target network's prediction for a given state, where the weighting is controlled using an exponential decay, defined in Equation 2:

$$\epsilon = \epsilon_{end} + (\epsilon_{start} - \epsilon_{end}) \cdot e^{-N/\epsilon_{decay}} \tag{2}$$

The values of $\varepsilon_{\text{start}}$, ε_{end}, and $\varepsilon_{\text{decay}}$ are hyperparameters which control the weight decay and N is the total number of training steps taken thus far across all epochs. We define the overall objective function of our system as seen in Equation 4. Borrowing terminology from Deep TAMER [5], we let $\mathcal{S}$ denote the set of *states* in which an agent can find itself and let $\mathcal{A}$ denote set the set of actions that this agent may execute. An agent selects and executes actions $(a_1, a_2, a_3, ...)$ to result in state trajectories $(s_1, s_2, s_3, ...)$. We denote the human feedback signal at each timestep as h_t which takes a continuous value between [-2, 2]. We assume that the human provides feedback according to some hidden function $H(\cdot, \cdot) :\rightarrow \mathcal{S} \times \mathcal{A}$ that the human implicitly understands.

The goal of learning problem is to compute an estimate of this function denoted as $\hat{H}$. Our target network and main network are paramaterized by θ and ϕ. Our loss function minimizes the difference in predicted human feedback, $\hat{H}_\theta(\mathbf{s}_t, a_t)$ with the given human feedback h_t (Equation 3). We then add Equation 4 which minimizes the difference between the predicted human feedback of each artificial state and a weighted sum of the true human feedback with the target network's predicted human feedback. The variable, c_t, denotes the weight of the sample used in TAMER [4], where state-action pairs are collected from a window in time before the state-action on which the human actually gives feedback. For our implementation, we choose a uniform distribution and save 10 of the previous state-action tuples for every feedback we receive from the human. ϵ denotes the exponential weight decay from Equation 2.

$$\mathcal{L}(\theta, \phi, (\mathbf{s}_t, a_t, h_t, c_t)) = c_t \Big[(\hat{H}_\theta(\mathbf{s}_t, a_t) - h_t)^2 + \tag{3}$$

$$\sum_{\mathbf{s}' \sim \mathcal{N}}^{k} \Big(\hat{H}_\theta(\mathbf{s}', a_t) - \Big[(1-\epsilon) \hat{H}_\phi(\mathbf{s}_t, a_t) + \epsilon h_t \Big] \Big)^2 \Big] \tag{4}$$

IV. Experimental Design

We conducted a 2 x 4 mixed subjects design to test our natural language processing (NLP) extension of Deep TAMER to baseline Deep TAMER across two different domains. We choose the domains as our between subjects factor and Deep TAMER variants as our within subjects factor. We identified a key issue in that while the user was giving feedback by speaking, the agent would continue to act. As the agent continued to act, it was not obvious to which state the feedback from the human should apply to. We therefore added a pausing condition to allow the human to precisely stop the agent when it required feedback, thereby allowing the agent to pay attention to the state-action pairs to which it is currently closest.

To normalize this confounding factor between groups, we added this *pausing* condition to both baseline Deep TAMER and also our NLP variant. We therefore have 4 conditions: (1) baseline Deep TAMER, (2) Deep TAMER with pauses, (3) NLP with Deep TAMER, and (4) NLP with Deep TAMER and pauses. In the TAMER with pauses condition, the simulation pauses for a few seconds every time the user provides feedback. In the NLP-with-pauses condition, the simulation of the agent pauses while the user is providing feedback and continues when the user finishes speaking. To match the original Deep TAMER paper exactly as our baseline, we only add the hypersphere augmentation and target network for the NLP conditions.

We tested these four Deep TAMER variants on two separate domains: mountain car [12] and Fetch-Reach [11] from OpenAI gym [18]. We recruited 18 participants, 12 trained the mountain car domain and 6 trained on the Fetch-Reach domain. An example of our experimental setup is shown in

Figure 1. Each participant was assigned randomly to either the mountain car or Fetch-Reach domain and trained the agent in all four Deep TAMER variants. The order that these variants were presented were randomly chosen. This experiment was conducted with the approval of the university's Institutional Review Board (IRB) under protocol H21268 and all subjects signed approved consent forms.

The experiment proceeded as follows: The participant first took the Negative Attitudes Towards Robots Scale (NARS) trust survey [19], a survey on their experience teaching others, and a big five personality test [20]. The experimenter then showed the participant how to train the agent and the participant practiced training for 10 iterations. The participant then trained the agent with all four Deep TAMER variants, taking a usability [21] and NASA TLX [22] workload survey after each training session.

V. Results

In this section, we report the results of the study and the statistical findings. We establish statistical significance at the $\alpha = 0.05$ level. All statistical analyses used data from 12 participants that completed mountain car [12] as the study was prematurely ended when none of the 6 participants in OpenAI's Fetch-Reach task [11] could train an agent to achieve reward. We note that for the mountain car task, solved is considered achieving at least -110 reward over 100 episodes. While no standard is set for the Fetch-Reach task, we consider solved as achieving at least -200 reward which represents the robot moving its gripper to the goal location and staying there for half the run length.

We conducted three linear mixed effects analyses for the relationship between usability, workload, and achieved agent scores with the Deep TAMER variants. As fixed effects, we used baseline trust, age, race, gender, and education. The baseline trust was determined by the pre-survey participants took before interacting with the agent. As random effects, we had intercepts for each participant. The residuals of the model were checked to be normally distributed by a visual inspection of the Q-Q plot.

Main effects were found for the Deep TAMER variant on usability $F(3, 33) = 4.270, p = 0.012$ and on the agent score $F(3, 33) = 2.815, p = 0.054$ (Figure 2). No main effects were found for the workload. Pairwise tests found that users scored Deep TAMER was significantly better than NLP with pauses ($p = 0.007$) for usability and baseline Deep TAMER agents significantly outperformed NLP agents ($p = 0.04$) on mean reward achieved from 20 runs.

VI. Lessons Learned

Our results showed these key findings: (1) While experimenters were able to train the both mountain car and fetch reach successfully with NLP variations of Deep TAMER, end-users were not; (2) Providing language commands that only provided semantic feedback resulted in a worse user experience and worse agent performance than the baseline keyboard Deep TAMER feedback.

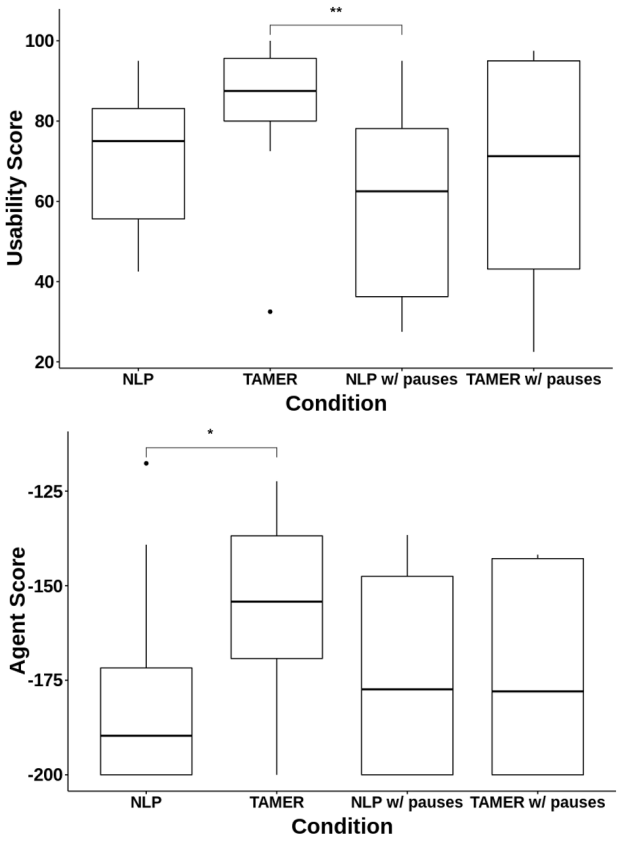

Fig. 2. **Rated Usability (top) and Learned Agent Scores (bottom) between the Deep TAMER Variants on Mountain Car** We show the results of our mixed-effect analysis here with $p < 0.05$ denoted by (*) and $p < 0.01$ denoted by (**). While the plot cannot show all the controlled effects, we still see base Deep TAMER rated higher than the other 3 variants

a) Experimenter-User Feedback Optimality Difference: We have several hypotheses on the difference between agent performances between experimenters and end-users. The experimenters were able to solve both environments as compared to 29% of the end-user runs for mountain-car and none of the end-user runs for Fetch-Reach. A policy trained just by Deep TAMER was not able to achieve satisfactory reward in Fetch-Reach even by the experimenters. This was most likely due to the added benefits of the artificial states and target network introduced in the NLP variants of Deep TAMER. In this work, we compared to baseline Deep TAMER but in future work, an ablation study could identify which of the three components (artificial states, target network, or language) helped experts the most in training agents. However, the ability for experimenters to train agents did not translate to end-users.

It is known that end-users provide suboptimal feedback [23]. In the Fetch-Reach case, we compare the experimenter's feedback and the user's feedback with an optimal policy. For the Fetch-Reach scenario, we define an optimal policy as any action that moves the gripper closer to the goal point in any state. We found that end-users gave almost two times as many suboptimal feedback instances as the experimenter. The suboptimal feedback for the end-users in the Fetch-Reach

task consisted of users giving positive feedback when the robot was moving away from the target and giving negative feedback when the robot moved towards the target. This difference in feedback between experts and end-users was compounded in our implementation of sampling feedback from nearby states, resulting in artificial states receiving the wrong feedback much more frequently when end-users provided feedback as compared to experimenters. This reinforces our challenges of using suboptimal feedback for training agents.

One open question is how to build a curriculum for *teaching users* how to teach robots. How many example training sessions should be shown? How do we convey the constraints of our system to the user? Additionally, are there any hyperparameter changes to our learning algorithms that differ between end-users and experts? In our implementation, the number of artificial states varied the performance of the learning algorithm between expert and end-users. Experts were able to use the artificial states to provide more optimal feedback during the agent's training process but when end-users gave suboptimal feedback, the large number of artificial states that were labeled with incorrect feedback reduced the performance of the agent. We show further research into how to account for suboptimal feedback from both algorithmic design to hyperparamater selection will be essential to allow end-users to successfully train learning agents,

b) Inadequate Linguistic Ability of Our Agent: Several issues were found related to our natural language model for the user. First, our language commands were only limited to positive or negative feedback and the system could not comprehend semantic information. For example, users could not tell the agent "move to the right" but only language relating to how well it was performing. This means that the primary beneficiaries from using language as a compressed signal of semantics was lost. Silva et al. [24] show one method on how the semantics of language can be used to accelerate imitation learning and reinforcement learning. Second, talking required much more effort from the users than the keyboard and the rate of feedback was also much slower. This was reflected on the usability scores and perceived workload scores. Users quickly grew tired of repeatedly providing vocal feedback.

We find that creating an NLP system for feedback but limiting to positive and negative sentiment limits our ability to convey meaningful instructions to agents. Open questions within this domain are on how to combine task agnostic language as feedback to learning agents.

VII. Conclusion

We have presented an addition to the Deep TAMER framework to enable users to train robotic agents through natural language. We train a Transformer based model to predict sentiment in language and use a Gaussian Hypersphere for data augmentation to account for the data sparseness when providing language feedback.

We report our results from a 2x4 mixed-subjects experiment and found that end-users could not match the same reward as policies learned from the experimenters. Additionally, we could not show that language feedback significantly lowered workload or increase usability with our algorithm. This indicates a need for further research into the type of feedback users prefer to train agents as well as the need to reduce the experimenter-user optimality difference.

References

[1] Matthew Gombolay, Xi Jessie Yang, Bradley Hayes, Nicole Seo, Zixi Liu, Samir Wadhwania, Tania Yu, Neel Shah, Toni Golen, and Julie Shah. Robotic assistance in the coordination of patient care. *The International Journal of Robotics Research*, 37(10):1300–1316, 2018.

[2] Rainer Bischoff, Johannes Kurth, Günter Schreiber, Ralf Koeppe, Alin Albu-Schäffer, Alexander Beyer, Oliver Eiberger, Sami Haddadin, Andreas Stemmer, Gerhard Grunwald, et al. The kuka-dlr lightweight robot arm-a new reference platform for robotics research and manufacturing. In *ISR 2010 (41st international symposium on robotics) and ROBOTIK 2010 (6th German conference on robotics)*, pages 1–8. VDE, 2010.

[3] Maya Cakmak and Leila Takayama. Towards a comprehensive chore list for domestic robots. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 93–94. IEEE, 2013.

[4] W. Bradley Knox and Peter Stone. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the Fifth International Conference on Knowledge Capture*, K-CAP '09, page 9–16, New York, NY, USA, 2009. Association for Computing Machinery.

[5] Garrett Warnell, Nicholas R. Waytowich, Vernon Lawhern, and Peter Stone. Deep TAMER: interactive agent shaping in high-dimensional state spaces. *CoRR*, abs/1709.10163, 2017.

[6] James MacGlashan, Mark K. Ho, Robert Loftin, Bei Peng, Guan Wang, David L. Roberts, Matthew E. Taylor, and Michael L. Littman. Interactive learning from policy-dependent human feedback. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2285–2294. PMLR, 06–11 Aug 2017.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[8] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[9] Yijiong Lin, Jiancong Huang, Matthieu Zimmer, Yisheng Guan, Juan Rojas, and Paul Weng. Towards more sample efficiency inreinforcement learning with data augmentation, 2019.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[11] Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research, 2018.

[12] Andrew William Moore. Efficient memory-based learning for robot control. Technical report, 1990.

[13] Samantha Krening, Brent Harrison, Karen M. Feigh, Charles Lee Isbell, Mark Riedl, and Andrea Thomaz. Learning from explanations using sentiment and advice in rl. *IEEE Transactions on Cognitive and Developmental Systems*, 9(1):44–55, 2017.

[14] Brent Harrison, Upol Ehsan, and Mark O. Riedl. Guiding reinforcement learning exploration using natural language, 2017.

[15] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[16] Tasmia Tasrin, Md Sultan Al Nahian, Habarakadage Perera, and Brent Harrison. Influencing reinforcement learning through natural language guidance, 2021.

[17] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language*

Processing, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[18] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[19] Dag Sverre Syrdal, Kerstin Dautenhahn, Kheng Koay, and Michael Walters. The negative attitudes towards robots scale and reactions to robot behaviour in a live human-robot interaction study. 01 2009.

[20] Beatrice Rammstedt and Oliver P. John. Measuring personality in one minute or less: A 10-item short version of the big five inventory in english and german. *Journal of Research in Personality*, 41(1):203–212, 2007.

[21] John Brooke. Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, 189, 11 1995.

[22] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.

[23] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[24] Andrew Silva, Nina Moorman, William Silva, Zulfiqar Zaidi, Nakul Gopalan, and Matthew Gombolay. Lancon-learn: Learning with language to enable generalization in multi-task manipulation. *IEEE Robotics and Automation Letters*, 7(2):1635–1642, 2022.