Assisting Humans in Human-Robot Collaborative Assembly Contexts through Deep Q-Learning

Garrett Modery
School of Computing
Montclair State University
Montclair, USA
moderyg1@montclair.edu

Weitian Wang*
School of Computing
Montclair State University
Montclair, USA
wangw@montclair.edu

2024 IEEE MIT Undergraduate Research Technology Conference (URTC) | 979-8-3315-3100-3/24/$31.00 ©2024 IEEE | DOI: 10.1109/URTC65039.2024.10937588

***Abstract*—Collaborative robots, affectionately referred to as "cobots," serve to function alongside their human counterparts to help them complete a specific task. This differs from traditional systems in which the machines are set about their own jobs and are often locked behind cages so as to prevent human access in favor of safety. By removing these walls and introducing collaborative systems, a new level of versatility and productivity is opened within the contexts that they are often employed. This focus of human-robot interaction has grown in recent years, and alongside it the topic of teaching and learning from demonstration has been investigated. Several methods of implementation for this topic have been developed, and while they are potentially effective, they still have gaps in versatility. Thus, we propose a different method of robot learning from demonstrations through the employment of deep Q-networks. These networks permit effective learning not only with human demonstration data, but also with direct feedback from the collaborating user. The proposed solution is experimentally implemented in real-world human-robot collaborative tasks. Preliminary results and analysis suggest the competitive performance of the proposed approach. Future work of this study is also discussed.**

***Keywords*—robotics, human-robot interaction, learning from human demonstrations, collaborative tasks, algorithm**

I. Introduction

Autonomous robotic systems have been rapidly developed and employed in many contexts as part of the rise of Industry 4.0 [1-3]. The primary focus in the context of robotics within Industry 4.0 is the development of smart factories equipped with multitudes of sensors and advanced robotics designed to enhance safety, cost efficiency, and productivity [4-6]. These advanced robotics are often employed not only as those with designated tasks to perform within their own areas, but also as collaborative robots designed to function alongside human workers. Several methods exist for the development of these systems so that they may collaborate appropriately. One such approach is the Teaching-Learning-Prediction-Collaboration (TLPC) framework [7]. It involves task demonstrations that the robot learns from, then predicts human intentions, and collaborates with its human partner accordingly. In this way, the desired behavior is learned rather than rigidly hardcoded in human-robot interaction (HRI).

There are a multitude of studies regarding the refinement and implementation of deep Q-learning models, which are built on the reinforcement learning paradigm [8]. Several methods exist that seek to improve training times and, of course, the agent's score and therefore performance in its task. Deep Q-learning is slow compared to other learning paradigms, requiring the agent to select and perform an action, examine the results, store that experience, and learn from previous ones. It is for this reason that there has been much focus on speeding along training times in order to enhance its feasibility.

Notably, a large area of application for deep Q-learning networks is that of simulated environments, such as Atari 2600 games [9, 10]. Simulations prove to be an excellent training ground for the learning agent due to the speed at which it may learn, being unburdened by any physical restrictions that may slow the process down. For instance, the model receives the frame data from the game and produces estimated Q-values from that data. Thus, it is hindered often only by the speed of the game itself, provided there is sufficient computational power for it and the training. Deep Q-learning for robotics differs in the speed of its execution. An agent is limited by the time that it takes to execute an action and receive feedback in a physical space due to time of movement, calculations from sensor data, etc.

It should be noted, however, that deep Q-networks are consistently applied in the field of robotics, despite potential delays in the learning process. To circumvent this issue, simulated environments are still often used. Gu et. al, for instance, made use of a simulated environment for their experiments prior to physical, real-world experiments in their deep learning task for robotic manipulation, which they note, in particular, enables faster comparisons of design choices [11]. We implement a similar approach in our work, which will be discussed later. Similarly, a study conducted by Mohanty et al. utilized deep Q-learning for mobile robot navigation [12]. Specifically, the design focused on the topic of path planning in an environment consisting of obstacles that must be avoided while reaching a goal point. Much like the previous study, a simulated environment in addition to a physical environment was used for the production of the agent and its learned experiences.

The training time of the agent is not simply a matter of convenience, but of necessity as well, particularly in collaborative robotics tasks. A method of addressing the latency of physical systems in collaborative deep reinforcement learning tasks is proposed by Ghadirzadeh [13]. This approach leans heavily on the concept of human intention prediction and determines the correct time for the robot to act and begin executing its actions based on gathered human motion data in addition to the current state of a developed behavior tree. It does so by integrating the Q-function as a node within the behavior tree itself to make its proactive decisions. In doing so, the model gains a "head start" on acting, cutting out some of the inevitable delays of physical systems.

In this work, we employ a deep Q-network to facilitate robot learning and enhance versatility in the TLPC framework in human-robot collaborative contexts. This solution allows the robot to explore its action space and receive direct feedback from its human partner for each action taken, thus "reinforcing" its task knowledge to further assist the human in collaborative tasks. By utilizing this method, the instruction of the robot shifts from static teaching to a dynamic relationship in which feedback can be consistently given to help the robot improve its behaviors.

II. Recap of TLPC Framework

As noted previously, this study extends the development of the TLPC framework [7], which is outlined in distinct segments as shown in Fig. 1. First, a human teammate teaches the robot how to perform a task. This step is notably minimally demanding of the user, simply requiring them to perform the task as they normally would. From this demonstration, the robot handles the learning process in which it takes this demonstration data and applies it in a way that permits it to build and optimize its task strategies. The learned knowledge permits the robot to predict human intentions in shared tasks, and as a result, to collaborate effectively with its human partner.

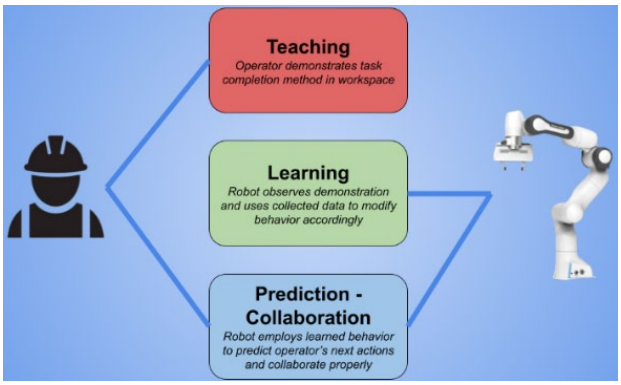

Fig. 1. The TLPC Framework.

III. Methodology

A. Deep Q-Learning

Reinforcement learning is a learning paradigm designed to permit an agent to learn the optimal policy for sequential decision problems [14]. This process involves the computation of the appropriate action value for a given policy [8]. The learning performance is improved with the development and refinement of this policy, which comes as a result of consistent exploration and exploitation within an agent's environment where the agent is the learner of the policy and executor of the actions, and the environment is the space that it is learning within.

The structure for a typical reinforcement learning setup is comprehensively introduced in [8]. In the learning process, at time *t*, the agent observes a state s_t from its environment (where s_t is in state space *S*) and performs an action a_t selected from action space *A*. This action selection is based upon the policy that defines the agent's behavior, usually denoted as π, which maps states to actions. Following the action, a reward of r_t is provided to the agent according to the quality of the action taken, and the current state transitions to the state s_{t+1} according to the state transition probability $P(s_{t+1}|s_t, a_t)$ [8]. Naturally, good actions produce a positive reward, and bad actions produce negative ones. This is defined by the reward function $R(s, a)$. The agent progresses through the transitions until a done state is reached, at which point a new episode is begun.

The goal of the agent is to gain the highest reward possible from its environment, and this process involves the estimation of the maximum Q-values possible for a given state and action. These values are derived from the Bellman equation [15]:

$$Q(s, a) = r(s, a) + \gamma \max Q(s', a'). \quad (1)$$

In Eq. (1), the Q-value for state *s* and action *a* is calculated by the immediate reward for taking action *a* in state *s* plus the maximum reward for the next state s' over all possible actions a'. The discount factor γ is applied to this to offset future rewards against immediate ones, where $\gamma \in (0,1]$ [8, 15].

Our implementation of deep Q-learning in the TLPC framework consists of two deep neural networks – a main and target network – as is typical for a double Q-learning setup [14]. The main network is used for the predictions of the Q-values for a given state, and the target network is used for the calculation of loss at each step [16]. These models are built with a combination of one-dimensional convolution layers, dense layers, and an LSTM layer, as our data is sequential in nature. The input state representation is a two-dimensional one-hot array of shape (*LT, NA*), where *LT* denotes the maximum length of a given task, and *NA* represents the number of possible actions. From this representation, the sequence of the task is effectively modeled for each state.

B. Modeling of Robot Learning for HRI

As shown in [16], the efficacy of pre-training the agent with demonstration data prior to giving it access to the environment is promising. With this, it is possible to effectively cut down on the training time in which a human is required to be present for explicit feedback. This pre-training phase makes use of four loss types for updates to the network: a one-step double Q-learning loss, an n-step double Q-learning loss, a supervised large margin classification loss, and finally an L2 regularization loss.

The one- and *n*-step double Q-learning loss are forms of temporal difference (TD) loss, and in our implementation, we use the Huber loss function which utilizes both mean squared error and mean absolute error, as defined below [15]:

$$\mathcal{L}(\delta, y, f(x)) = \begin{cases} \frac{1}{2}(f(x) - y)^2 & \text{if } |f(x) - y| \le \delta \\ \delta|f(x) - y| - \frac{1}{2}\delta^2 & \text{if } |f(x) - y| > \delta \end{cases}, \quad (2)$$

where $\mathcal{L}$ represents the Huber loss function, δ represents the delta parameter that establishes the threshold for switching between the two components of the function, *y* is the target value, and *f(x)* is the predicted value. Implementing *n* step returns helps to propagate the expert's trajectory values to earlier states, thus producing more effective pre-training [16]. The returns are calculated as:

$$r_t + \gamma r_{t+1} + \gamma^{n-1} r_{t+n-1} + \max_a \gamma^n Q(s_{t+n}, a), \quad (3)$$

where r_t is the reward at timestep *t*, γ is the discount factor, *n* is the number of steps into the future that are being considered, and the final part of the equation determines the maximum estimated Q-value for the state at timestep *t+n* for all actions *a*.

979-8-3315-3100-3/24/$31.00 ©2024 IEEE

The large margin classification loss is crucial to the pre-training phase of the algorithm, as it forces non-demonstrator actions to be at least a margin lower than those taken by the expert [16]. In other words, this encourages the network to prioritize demonstrator actions above others. The loss is calculated as:

$$J(Q) = \max[Q(s,a) + l(a_E, a)] - Q(s, a_E), \quad (4)$$

where a_E is the action of the demonstrator, and $l(a_E, a)$ is the margin function that is positive if $a \neq a_E$, and 0 if $a = a_E$.

A simulated environment is also utilized to enable faster training of the deep Q-network. This environment offers the benefits of allowing the agent to take actions and observe the results much faster than it could from direct physical interactions. The agent explores actions that it may take based upon what is available in the action space. It is penalized for nonsensical actions, such as attempting to select an assembly piece that has already been taken, and rewarded for proper ones that align with the demonstrated data. These transitions are organized in the replay buffer, and each time it is sampled to train the model, a random subset of the demonstration data is inserted into it as they are prioritized experiences. By training the model in this way, it becomes more robust and less likely to act improperly when physically interacting with the human.

In real-world HRI, the human's part in teaching the robot involves more processes. Small adjustments may need to be made to the agent's behavior that the simulated training may not have accounted for. Thus, the collaborative phase of the HRI may still permit learning to take place, if necessary. In this work, the human teaching and robot learning algorithm is shown in Algorithm 1, which has been adapted from [16]. The definitions of the symbols used in Algorithm 1 are δ_d: expert-demonstrated assembly task transitions data set, δ_r: replay buffer, θ_t: target network weights, θ_m: main network weights, E_{sim}: simulated environment, E_{phy}: physical environment, k: pre-training steps, μ: target model update frequency, τ_{sim}: simulated environment training steps, τ_{phy}: physical environment training steps, and π: behavior policy.

Algorithm 1 Human teaching and robot learning for human-robot collaborative tasks through deep Q-learning

"Teaching phase" in which assembly task methods are demonstrated to the robot, which are organized as transitions in δ_d. δ_d is used to fill δ_r
Main and target models are initialized with Xavier uniform initializer and 1D convolution layers, an LSTM layer, and Dense layers
Simulated environment E_{sim} that mimics physical environment E_{phy} is initialized
for i = 0 to k
  Sample δ_r for a batch of n transitions
  Use the target model and main model to compute the loss
  Use the calculated loss to perform gradient descent and update the weights of the main model θ_m
  if i **mod** μ = 0 **then**
    $\theta_t \leftarrow \theta_m$
  end if
end for
for i = 0 to τ_{sim}
  Select action from epsilon-greedy policy π
  Perform action in E_{sim} and examine result (s', r, d)
  Place transition (s, s', r, a, d) into replay buffer δ_r. Replace oldest transition if capacity is reached
  Sample δ_r for a batch of n transitions
  Use the target model and main model to compute the loss
  Use the calculated loss to perform gradient descent and update the weights of the main model θ_m
  if i **mod** μ = 0 **then**
    $\theta_t \leftarrow \theta_m$
  end if
end for
for i = 0 to τ_{phy}
  Select an action from predicted good actions
  Perform action in E_{phy} and examine result (s', r, d)
  Place transition (s, s', r, a, d) into replay buffer δ_r. Replace oldest transition if capacity is reached
  Sample δ_r for a batch of n transitions and δ_d for a mini-batch of transitions
  Use the target model and main model to compute the loss
  Use the calculated loss to perform gradient descent and update the weights of the main model θ_m
  if i **mod** μ = 0 **then**
    $\theta_t \leftarrow \theta_m$
  end if
end for

IV. Experimental Design

As shown in Fig. 2, the experimental setup for our human-robot collaborative experiment utilizes several components. The platform being used is a Franka Emika Panda robot with seven degrees of freedom to enable greater flexibility of movement [17, 18]. Fifteen letter blocks are placed within the workspace that represent assembly parts. These parts may be selected from the workspace in the user's desired order and length. This permits a greater degree of task customization for the participant than what has been offered in our previous studies, and we believe that this improvement will enhance the quality of the collaboration. Additionally, the robot is equipped with an Intel RealSense D435i depth camera mounted to the end effector that allows it to view the workspace. The code that executes Algorithm 1 is run from a Lenovo P520 ThinkStation and utilizes the MoveIt Commander Python API to plan and execute trajectory commands [19, 20]. Those trajectories are calculated using deprojected points from the RealSense camera, which has a YOLOv8 model trained on the assembly parts performing labeling on its collected frames.

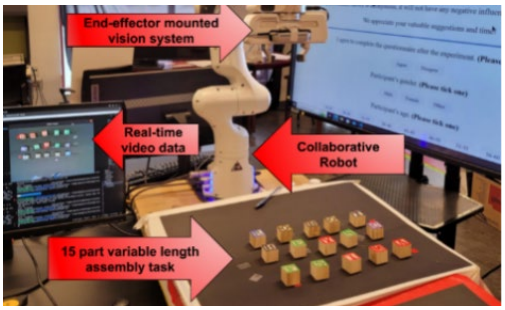

Fig. 2. Experimental setup.

979-8-3315-3100-3/24/$31.00 ©2024 IEEE

The task design of our human-robot collaborative experiment is outlined in Fig. 3. The human first demonstrates three assembly sequences to the robot, which collects this data with an end-effector-mounted vision system. The demonstration data is pre-processed to transitions and organized within a priority replay buffer separate from the standard one. This completes step 1 of Algorithm 1. Before the assembly process begins, steps 2-22 of Algorithm 1 are run to prepare the model for physical human-robot interaction. The next steps shown in Fig. 3 are representative of steps 23-33 in Algorithm 1. The human takes the first part of their desired assembly sequence, which provides a starting point for the robot's prediction and collaboration. This process of prediction and collaboration aligned with the final stage of the TLPC framework continues until the task is completed. Actions taken in this phase produce transitions that are saved in the replay buffer and trained on, reinforcing the user's desired behavior of the robot.

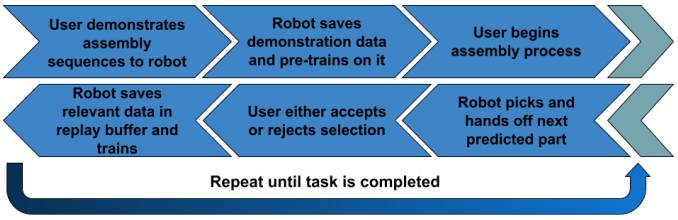

Fig. 3. Experimental task design.

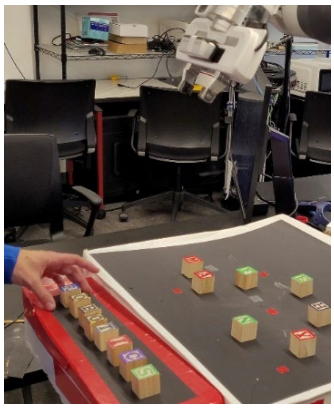

Fig. 4. The participant demonstrates the word ROBOTICS to the collaborative robot.

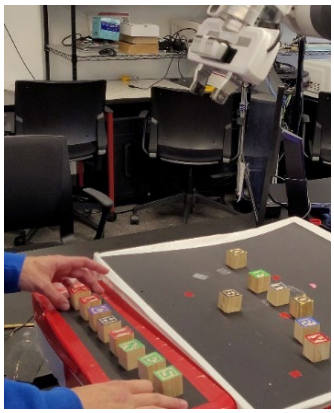

Fig. 5. The participant demonstrates the word MACHINES to the collaborative robot.

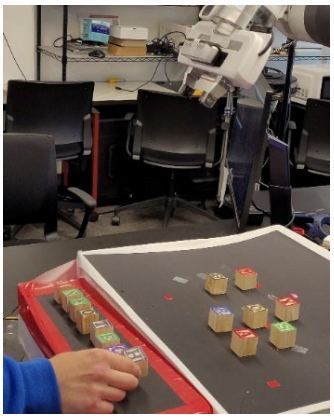

Fig. 6. The participant demonstrates the word INFOTECH to the collaborative robot.

Figs. 4, 5, and 6 present the human teaching phase of the above diagram (Fig. 3), in which the participant demonstrates assembly sequences to the collaborative robot. The human uses the letter blocks to assemble three words (ROBOTICS (Fig. 4), MACHINES (Fig. 5), and INFOTECH (Fig 6)) representing three different products according to working preference. The end-effector-mounted camera observes the selection space while the parts are assembled in the desired order to build its prerequisite task knowledge.

V. Results and Analysis

A. Learning Performance and Analysis

Preliminary testing of this framework has yielded promising results. The training and execution performance of the model is shown in Fig. 7, which outlines the time taken for policy convergence for three tested sequences: ROBOTICS, MACHINES, and INFOTECH. The agent receives a reward of 1 for appropriate actions, and a reward of -1 for those that are wrong. Actions are taken for the transition between letters, and thus a maximum reward of n-1 may be gained for a word of length n. The selected words above are each of length 8, establishing a "perfect score" as 7.

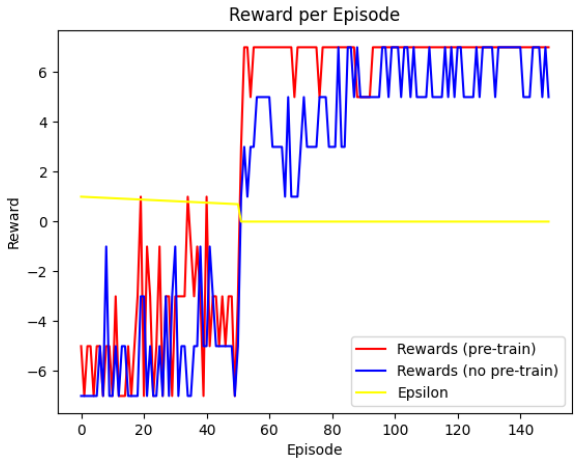

Fig. 7. Training and execution results.

The above results present the reward per episode both for a pre-trained model (red) and for a non-pre-trained model (blue)

979-8-3315-3100-3/24/$31.00 ©2024 IEEE

for 150 episodes. Additionally, the epsilon parameter is graphed as well, which begins at the max value of 1.0 and decreases to the minimum value of 0.5 as a function of,

$$\varepsilon = \max(\varepsilon_{max} - \zeta \times e,\ \varepsilon_{min}), \tag{5}$$

per episode, where the current episode number is denoted by e, and ζ represents the epsilon decay rate, defined as,

$$\zeta = \frac{(\varepsilon_{max} - \varepsilon_{min})}{e_{total}}, \tag{6}$$

where, e_{total} represents the total number of episodes that are being run. This ensures a uniform reduction of the epsilon value as exploration progresses and encourages slightly more exploitation as time goes on. It should be noted that epsilon is forced to zero at episode 50, which is also where both models begin to show improvements in their per-episode rewards. This is to be expected, as every action being taken is that which is deemed best. By episode 100, the model that had been pre-trained on the demonstration dataset had converged to the optimal policy and received consistent results. This is in contrast to the non-pre-trained model which fails to do so consistently, demonstrating the clear advantage of the developed solution, particularly in this context where humans either cannot or simply do not want to train the robot for an excessive amount of time.

It should also be noted that, due to the finite steps of the assembly tasks, the model is capable of converging on the optimal policy rather quickly. Given more complex tasks or even a non-expert demonstrator, the model's performance would undoubtedly be impacted and thus demand longer training times to converge. Additionally, we note that this framework may have trouble with similar demonstrated sequences that start the same way, as in this instance there would be several "optimal" actions that may be taken, and as a result, the wrong part may be predicted. In this scenario, however, the user would simply reject the part, and the next best one would be offered instead.

B. *Implementation in Real-World Human-Robot Collaborative Contexts*

Figs. 8, 9, and 10 present the human-robot collaborative process in action for the words ROBOTICS, MACHINES, and INFOTECH. The human demonstrations for each word in the selection area are shown in steps Figs. 8(1a), 9(1b), and 10(1c), respectively. Following this demonstration, the model is pre-trained with the collected data. During this time, the parts are placed back within the workspace and the user is instructed to take the first of the parts of any of the demonstrated sequences. Once the training is completed, the robot will examine the selection area to determine the current state and will proceed to make a prediction for the participant's next desired part and move to acquire that part for them. This is represented in step 2 of all relevant Figs. Following the part acquisition, the collaborating human will decide if that part is what they require or not. If it is, they will add it to the selection area and the robot will recognize that this was a good decision, and the agent will receive a positive reward. This is shown in steps 3a and 3b. Conversely, should the model make a bad prediction and the robot attempt to hand off the incorrect part, as shown in step 3c, the participant may simply add the part back to the workspace so that the agent can learn that this was a poor choice and may try again. This process of collaboration continues as seen in all stages through 6 until the task is ultimately complete.

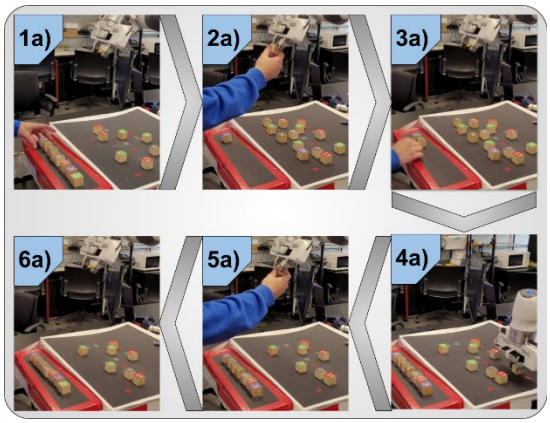

Fig. 8. Human-robot co-assembly for the "ROBOTICS" word.

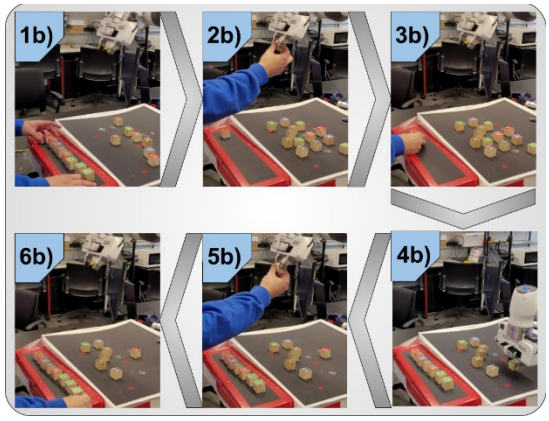

Fig. 9. Human-robot co-assembly for the "MACHINES" word.

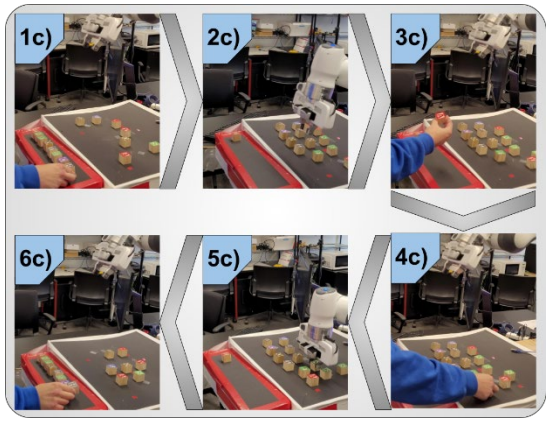

Fig. 10. Human-robot co-assembly for the "INFOTECH" word.

VI. Conclusions and Future Work

In this study, we have established a framework for human teaching and robot learning in HRI through deep Q-learning. The developed approach allows effective robot learning not only with human demonstration data, but also with direct feedback from the collaborating user. We have experimentally implemented the proposed solution in real-world human-robot collaborative tasks. Results and analysis suggest the competitive performance of the developed approach.

979-8-3315-3100-3/24/$31.00 ©2024 IEEE

Based on the experimental setup, we plan to further evaluate the performance of the proposed approach with multiple metrics to be gathered from the opinions of participants of the study. These metrics will be used to identify areas of strengths and weaknesses in the approach as well as user comfort, trust, and acceptance of the system. This data will be employed to make improvements to the framework as well as to gather insights into how varying demographics view it. In addition, the evaluation results will be compared against those which were previously collected for our prior study in order to determine which method better satisfies the TLPC framework.

We believe that the improvement in the maximum complexity of the task from our previous study will be a major contributing factor to the foreseen improvements in user ratings of the experience. This complexity serves to enhance the user experience by providing further customizability of the task, while also simplifying their efforts in teaching the robot. It is important to note that this complexity is entirely at the user's discretion and is not designed to overcomplicate the process, but rather to allow participants to demonstrate tasks according to their comfort or trust levels [21]. As such, we anticipate higher performance of human-robot teams, especially in the context of Industry 5.0, which will be investigated in our future study.

ACKNOWLEDGMENT

This work is supported in part by the National Science Foundation under Grants CMMI-2138351 and CMMI-2338767.

REFERENCES

[1] K.-D. Thoben, S. Wiesner, and T. Wuest, "Industrie 4.0" and smart manufacturing–a review of research issues and application examples," *Int. J. Autom. Technol,* vol. 11, no. 1, pp. 4-16, 2017.

[2] W. Wang, R. Li, Y. Chen, Y. Sun, and Y. Jia, "Predicting Human Intentions in Human-Robot Hand-Over Tasks Through Multimodal Learning," *IEEE Transactions on Automation Science and Engineering,* vol. 19, no. 3, pp. 2339-2353, 2022.

[3] W. Wang, R. Li, Z. M. Diekel, Y. Chen, Z. Zhang, and Y. Jia, "Controlling Object Hand-Over in Human–Robot Collaboration Via Natural Wearable Sensing," *IEEE Transactions on Human-Machine Systems,* vol. 49, no. 1, pp. 59-71, 2019.

[4] M. Javaid, A. Haleem, R. P. Singh, and R. Suman, "Substantial capabilities of robotics in enhancing industry 4.0 implementation," *Cognitive Robotics,* vol. 1, pp. 58-75, 2021.

[5] E. Herrera, M. Lyons, J. Parron, R. Li, M. Zhu, and W. Wang, "Learning-Finding-Giving: A Natural Vision-Speech-based Approach for Robots to Assist Humans in Human-Robot Collaborative Manufacturing Contexts," in *2024 IEEE 4th International Conference on Human-Machine Systems (ICHMS)*, 2024: IEEE, pp. 1-6.

[6] W. Wang, R. Li, Z. M. Diekel, and Y. Jia, "Robot action planning by online optimization in human–robot collaborative tasks," *International Journal of Intelligent Robotics and Applications,* vol. 2, no. 2, pp. 161-179, 2018.

[7] O. Obidat, J. Parron, R. Li, J. Rodano, and W. Wang, "Development of a Teaching-Learning-Prediction-Collaboration Model for Human-Robot Collaborative Tasks," in *2023 IEEE 13th International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)*, 11-14 July 2023 2023, pp. 728-733.

[8] Y. Li, "Deep Reinforcement Learning," *CoRR,* vol. abs/1810.06339, 2018. [Online]. Available: http://arxiv.org/abs/1810.06339.

[9] I.-A. Hosu and T. Rebedea, "Playing atari games with deep reinforcement learning and human checkpoint replay," *arXiv preprint arXiv:1607.05077,* 2016.

[10] V. Mnih *et al.*, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[11] S. Gu, E. Holly, T. P. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation," *arXiv preprint arXiv:1610.00633,* vol. 1, p. 1, 2016.

[12] P. K. Mohanty, A. K. Sah, V. Kumar, and S. Kundu, "Application of Deep Q-Learning for Wheel Mobile Robot Navigation," in *International Conference on Computational Intelligence and Networks (CINE)*, 2017, pp. 88-93.

[13] A. Ghadirzadeh, X. Chen, W. Yin, Z. Yi, M. Björkman, and D. Kragic, "Human-Centered Collaborative Robots With Deep Reinforcement Learning," *IEEE Robotics and Automation Letters,* vol. 6, no. 2, pp. 566-571, 2021.

[14] H. a. G. van Hasselt, Arthur and Silver, David, "Deep Reinforcement Learning with Double Q-Learning," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 30, 2016, doi: 10.1609/aaai.v30i1.10295.

[15] M. Shaili and A. Anuja, "Double Deep Q Network with Huber Reward Function for Cart-Pole Balancing Problem," *International Journal of Performability Engineering,* vol. 18, no. 9, pp. 644-653, 2022.

[16] T. Hester *et al.*, "Deep Q-learning From Demonstrations," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 32, no. 1, Apr. 2018.

[17] I. Jacoby, J. Parron, and W. Wang, "Understanding Dynamic Human Intentions to Enhance Collaboration Performance for Human-Robot Partnerships," in *2023 IEEE MIT Undergraduate Research Technology Conference (URTC)*, 2023: IEEE, pp. 1-6.

[18] C. Hannum, R. Li, and W. Wang, "A Trust-Assist Framework for Human-Robot Co-Carry Tasks," *Robotics,* vol. 12, no. 2, pp. 1-19, 2023.

[19] H. Diamantopoulos and W. Wang, "Accommodating and Assisting Human Partners in Human-Robot Collaborative Tasks through Emotion Understanding," in *2021 International Conference on Mechanical and Aerospace Engineering (ICMAE)*, 2021: IEEE, pp. 523-528.

[20] S. Bier, R. Li, and W. Wang, "A Full-Dimensional Robot Teleoperation Platform," in *2020 IEEE International Conference on Mechanical and Aerospace Engineering*, 2020: IEEE, pp. 186-191.

[21] J. Parron, T. T. Nguyen, and W. Wang, "Development of A Multimodal Trust Database in Human-Robot Collaborative Contexts," in *2023 IEEE 14th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2023: IEEE, pp. 0601-0605.