2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Context-Aware Conversation Adaptation for Human-Robot Interaction

Zhidong Su and Weihua Sheng*

Abstract— **Existing conversational robots are mostly reactive in that the interactions are usually initiated by the users. With the knowledge of the environmental context such as people's daily activities, robots can be more intelligent and proactive. In this paper, we proposed a context-aware conversation adaptation system (CACAS) for human-robot interaction (HRI). First, a context recognition module and a language processing module are developed to obtain the context information, user intent and slots, which become part of the state. Second, a reinforcement learning algorithm is developed to train an initial policy with a simulated user. User feedback data is collected through HRI using the initial policy. Third, a policy combining the reinforcement learning-based policy with the neural network-based policy is adapted based on the user feedback. We conducted both simulated user tests and real human subject tests to evaluate the proposed system. The results show that CACAS achieved a success rate of 85% in the real human subject test and 87.5% of participants were satisfied with the adaptation results. For the simulation test, CACAS had the highest success rate compared with the baseline methods.**

I. Introduction

Social robots are usually equipped with cameras and microphones, which allow the development of visual and auditory abilities. Social robots can recognize the environmental context of a user, such as activities of daily living (ADLs), user's facial expression, home scene, etc. For example, existing social robots like Jibo [1], ElliQ [2] and Amazon Astro [3] have cameras and microphones. They can take photos and recognize human faces. These robots are also able to conduct natural language conversation with humans to perform tasks such as online shopping, checking weather, telling jokes and news, creating reminders [4], etc. Even these functions are useful and important in human's daily life, the separation of natural language conversation and the awareness of the environmental context makes the robots passive and unable to adapt the conversation to the users' situations and needs. For example, Do *et al*. [5] utilized a social robot to recognize human daily activities through the sound collected from the home environment. However, how the detected daily activity information can help improve the robot's intelligence and provide better services is not considered.

Knowing the environmental context such as what the human is doing is important to human-robot interaction (HRI). The works in [6]–[8] enable robots to understand human actions and their environments and utilize the sensing results to determine the corresponding actions. However, such context information is not utilized for the robot to initiate a conversation or the adaptation of the HRI to better serve the humans. We believe that it would be beneficial to improve robot intelligence if the robot can proactively start the conversation and adapt to user's preferences in different environmental contexts.

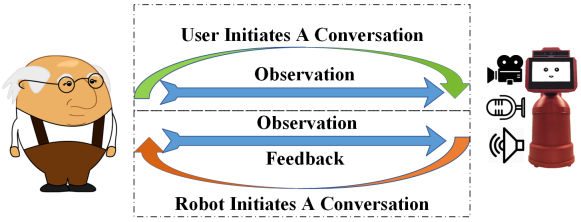

Fig. 1: The context-aware conversation adaptation system.

In this paper, we proposed a context-aware conversation system for HRI that is both proactive and adaptive. As shown in Fig. 1, the robot uses its microphone and camera to observe the user's activity and initiates a conversation if needed. Based on the user feedback and the observed context, a reinforcement learning (RL)-based method is proposed to adapt the robot's actions. In addition, a user-initiated conversation is utilized as extra feedback for the adaptation. This method can optimize the system performance gradually based on the responses or feedback from the users.

The main contributions of this paper are four folds. First, the proposed CACAS enables a robot to proactively initiate conversations and adapt to users in particular contexts, which extends the frontier of HRI research. Second, we proposed a user-modeling method that enables the robot to adapt its actions in a simulation training environment which requires less user effort, therefore making it more practical for robot deployment in the real world. Third, we combined the RL-based policy with a supervised learning-based policy to take advantage of the strength of both policies. Fourth, we conducted both simulation tests and human subject tests to evaluate the performance of CACAS. The obtained results are very promising. Our proposed method can achieve an 85% success rate after 7 days' interaction and 87.5% of participants were satisfied with the adaptation results. Besides, CACAS also has the best performance in the simulation test, which is 19%, 36%, and 68% higher than the three baseline methods.

The rest of this paper is organized as follows. Section II introduces the related work. Section III gives an overview of

This project is supported by the National Science Foundation (NSF) Grants CISE/IIS 1910993, EHR/DUE 1928711, CPS 2212582, and TI 2329852.

Zhidong Su and Weihua Sheng (corresponding author) are with the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK, 74078, USA (e-mails: zhidong.su@okstate.edu, weihua.sheng@okstate.edu).

the system. Section IV details the proposed method. Section V gives the experimental results. Section VI concludes our work and discusses the future work.

II. Related Work

Context information like daily activities and locations in the environment is useful in human robot interactions. It can help reduce the uncertainty in human intent recognition, and offer an additional information channel for better interaction. Trick *et al.* [6] integrated speech, gesture, gaze direction and scene objects for improved HRI. The obtained context information is used to reduce the uncertainty about the user intention. However, if the context or user preference changes, their method is unable to handle it. Do *et al.* [5] proposed a sound-based human activity monitoring framework for social robots. However, how the activity helps users in HRI is not considered. Giorgio *et al.* [9] proposed a framework which extracts the contextual information from the environment and builds a context reasoning module to translate the contextual knowledge into the robot's behaviors. However, even equipped with the context-aware ability, most of the existing social robots did not use the context to adapt pre-defined functions to different users in different contexts.

Tapus *et al.* [10] proved that if the robot behavior fits user's preference or personality, the users tend to spend more time with the robot. Ritschel *et al.* [11] adapted the robot's linguistic style based on the social signals like engagement or no engagement. Park *et al.* [12] used a model-free RL method to adapt the robot's story telling strategy based on the students' performance. Tsiakas *et al.* [13] personalized the cognitive training of a socially-assistive robot based on the task engagement and the user performance during the task. To solve the sparse reward problem, Ferreira *et al.* [14] utilized a socially-inspired reward obtained from human emotion as an extra reward to speed up the adaptation process. Their work can adapt the robot model to different user features/personalities. However, how to adapt to users in particular contexts and when to deploy a trained model to different users, how to adapt its actions using less user input are not addressed.

III. System Overview

The software architecture of the proposed context-aware conversation adaptation system is shown in Fig. 2, which has three parts: a context recognition module that can recognize human daily activities based on sound and home scenes through images; a language processing module that processes user utterance and generates response audio, which is implemented in the same way shown in our previous work [4]; and an adaptation module that personalizes the dialog agent based on the human-robot interactions and the context.

Context Recognition. The context information provides extra information source in addition to the user speech, which enables the robot to perform more proactively and intelligently. In this paper, we focus on the sound-based daily activities and image-based home scenes. For the sound-based daily activities, there are 33 home environment

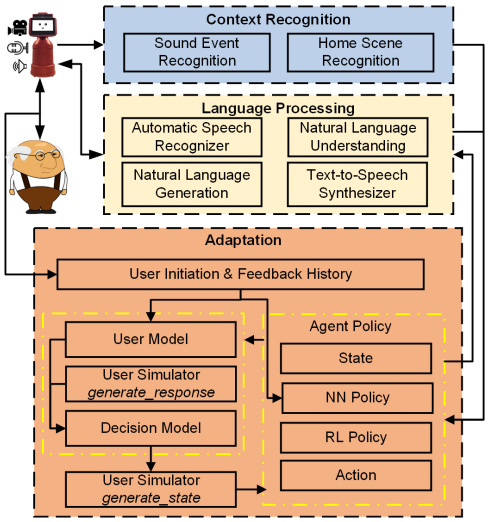

Fig. 2: The software architecture of the proposed CACAS.

sound events considered, namely, 1: *cough*, 2: *cry*, 3: *clear throat*, 4: *sneeze*, 5: *sniff*, 6: *fall down*, 7: *burp*, 8: *yawn*, 9: *snore*, 10: *drink water*, 11: *drink milk*, 12: *drink soup*, 13: *eat apple*, 14: *eat noodle*, 15: *eat chips*, 16: *wash hands*, 17: *wash clothes*, 18: *wash dishes*, 19: *use scissor*, 20: *use blender*, 21: *use stove*, 22: *use hair dryer*, 23: *use microwave*, 24: *use fan*, 25: *watch TV*, 26: *type keyborad*, 27: *cut food*, 28: *fry food*, 29: *pour water into glass*, 30: *shave*, 31: *brush teeth*, 32: *fulsh toilet*, 33: *do nothing*. For the image-based home scenes, we consider 6 categories, namely, 1: *bedroom*, 2: *living room*, 3: *kitchen*, 4: *bathroom*, 5: *dining room*, 6: *others*. Utilizing the integrated microphone and camera, the robot can obtain the context of human activities. The obtained context is used as a part of the input state of the adaptation module.

Adaptation Module. The adaptation module adapts the robot actions using the reinforcement learning algorithm based on user feedback, proactive input and context, which is the main focus of this work.

Before real-world deployment of the robot, a user simulator is designed to interact with the robot to train an initial robot action policy. The reason to use the user simulator is that it is unrealistic to ask real users to interact with the robot from the beginning, as it requires many rounds of interactions which take a very long time.

After deploying the initial policy to the robot, the user feedback and user-initiated input can be collected through human-robot interactions. The user-initiated input and the user feedback are the user data that can be used to train a user model and a neural network (NN) based policy. The user model and the user simulator interact with the robot action policy to update the robot action policy. The adapted robot action policy is a combination of an RL-based policy and an NN-based policy. Both the user model and the user simulator can generate a response. Therefore a decision model is used to choose a proper user response.

IV. METHODOLOGY

In this section, we firstly introduce the context recognition algorithm that recognizes the activities based on sound and home scenes based on images. Secondly, the context-aware HRI is modeled as a reinforcement learning problem. The adaptation method is detailed at the end.

A. Context Recognition

We use the robot camera to capture home scene images and the microphone to record the sounds generated from human daily activities. For home scene recognition, we used the Residual neural network (ResNet) [15]. For sound-based activity recognition, we designed a Convolutional Neural Network (CNN). The Mel-frequency cepstral coefficient (MFCC) [16] is used as the sound feature. The duration of the sound sample is set to 1 second. The sample rate is 32,000 Hz. The window size of the fast Fourier transform is set to 2,048 and the step size is 1,024. The number of the Mel band is set to 64. For each sound sample, the generated feature shape is 64x32. We built a CNN network that has 3 convolutional layers (kernel size: 3 x 3, dimension: 64, 128, 256) and 1 dense layer (dimension: 128) to recognize the sound activity.

B. Context-Aware HRI Formulation

1) HRI Process Modeling: The robot performs an action by observing the current context and the interaction history, which can be regarded as a Markov Decision Process (MDP) [17]. The MDP can be described as a tuple $\{S, A, T, R, \gamma\}$. S is the state set. The state $s \in S$ includes the environmental context and dialog status, which is the input to the dialog agent/robot model. A is the action set, which is the output of the dialog agent model. T is the state transition probability matrix, which is usually used in the model-based reinforcement learning method. R is the reward function. $\gamma \in$ [0,1] is the discount factor. The cumulative reward with the discount factor is $R = \sum_{t=0}^{+\infty} \gamma^t r_t$, where t is the time step and $r_t = R(s_t, a_t)$. The goal of MDP is to derive an optimal policy which maximizes the cumulative reward. The policy function π is a probability density function and maps the state to an action $\pi : S \rightarrow A$.

Table I lists the different components and the corresponding dimensions of the state and the agent actions. We utilize the one-hot vector to represent different state components and concatenate them together to form the state. The number in the brackets shows the dimension of each component. The sub-state S_1 and S_2 were explained in the $System\ Overview$ section. The sub-state S_3 is the time when the activity is observed. We divide the whole day into 12 segments of 2 hours each. Therefore, $S_3 = \{[0,2), [2,4), [4,6), [6,8), [8,10), ..., [20,22), [20,24)\}$. The sub-state S_4 is the time between the current activity and the previous activity. The $do\ nothing$ activity is excluded here. S_4 is defined as $S_4 = \{0, 1, 2, 5, 10, 15, 20, 30, 60, 90, 120, 200\}$. The unit of these values is minute. S_5 indicates the task that the robot is performing, which is the same as the agent action. The sub-state $agent\ action$ is the action that the robot needs to take based on the current observation. We designed a total of 14 agent actions as shown in Table I. Those actions are designed to handle the daily activities and show the robot care for users. The sub-state $user\ action$ is defined as $S_7 = \{\text{“}accept\text{”}, \text{“}reject\text{”}, \text{“}others\text{”}\}$. The total state dimension is 94 and the agent action dimension is 14. The total number of state is 16,765,056. All sub-states are utilized by the robot policy to select an action.

The reward function provides a score to the dialog agent which indicates how well the agent behaves. The dialog agent's behavior is optimized based on the reward. In this study, if the user feedback intent is “accept”, reward R = 1. If the user feedback intent is “reject”, R = -1. Otherwise, R = 0.

TABLE I: State and agent actions.

State: S_1: sound-based daily activity (33), S_2: image-based home scene (6), S_3: current time (12), S_4: activity repeat time interval (12), S_5: current task (14), S_6: agent action (14) S_7: user action (3). **Agent actions**: do nothing, show compassion, chitchat, remind to take medicine, ask to offer help, call someone for help, set a timer, pain check, confirm sound, record sound, remind to turn off, remind to be careful, remind to put back, remind to unplug

2) Robot Action Policy Learning: We use the Soft Actor-Critic (SAC) algorithm [18] as the basic RL algorithm to approximate the optimal robot action policy. Different from model-based methods which require the state transition probability matrix T, SAC is a model-free policy-based RL method, which does not need to model the complex real world and is more sample-efficient than other policy-based RL methods such as TRPO [19] and PPO [20] because it is off-policy. There are three networks that need to be optimized during training, two Critic networks $Q_{\omega 1}(s,a), Q_{\omega 2}(s,a)$, and one Actor network $\pi_\theta(s)$. Eq. (1) shows the loss function to optimize the Critic networks, where N is the number of the training samples and y_i in Eq. (2) is the target value. γ and α are the hyperparameters. Eq. (3) shows the loss function to optimize the Actor network. After training, the Actor network is used to select an action that has the maximum probability as shown in Eq. (4).

$$L_{critic}(\omega_j) = \frac{1}{N}\sum_{i=1}^{N}(y_i - Q_{\omega_j}(s_i, a_i))^2, j = 1, 2 \quad (1)$$

$$y_i = r_i + \gamma \min_{j=1,2} Q_{\omega_j}(s_{i+1}, a_{i+1}) - \alpha log \pi_\theta(a_{i+1}|s_{i+1})) \quad (2)$$

$$L_{actor}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(\alpha log \pi_\theta(a_i|s_i) - \min_{j=1,2} Q_{\omega_j}(s_i, a_i)) \quad (3)$$

$$a = \arg\max_{a \in A}(\pi_\theta(s)) \quad (4)$$

3) User Simulator: To train a SAC model, it is not practical to employ real users from the very beginning to interact with the robot to obtain sufficient tuples/samples, since it is time- and resource-consuming. Therefore, we proposed a user simulator, which can reduce the human effort to train and adapt the robot action policy. We firstly constructed a basic daily activity-agent action acceptance (D4A) pairs. The D4A pairs show the user's sentiment polarity towards the agent action when performing a specific activity. On top of the D4A pairs, it is the user preferences. In response generation, the user preference has the first priority and the D4A pairs have the second priority. A function named $generate\_response$ is used to generate a response. It first checks if the current state s and agent action a fit the user preference. If it does not fit, D4A pairs are used to generate a response.

When using the user simulator to train the dialog agent, the user simulator generates a state every K minutes using function $generate\_state$, which utilizes the time, interaction history and user preference up to generate all sub-states. The time starts at 00:00 and ends at 23:59 and increases by K minutes each step. Therefore, there will be 24 x 60/K epochs in one day. For training an initial policy, up is used to generate sub-state elements. For adaptation training, the user data D is used to generate the state, in which the activity recorded in D has a higher probability of being selected.

C. Context-Aware Conversation Adaptation

The robot action policy trained by the initial user simulator cannot accommodate the preferences of all users. For example, a young user may not want to be offered as much help as an older person. Therefore, it is necessary to adapt the robot action policy to particular users. In the CACAS, the robot can proactively initiate a conversation based on its understanding of the user and the context. The users can also initiate a conversation if they need any help. We use the user-initiated input and feedback on the robot actions to model the user preference. In Fig. 2, the "Adaptation" module shows the proposed adaptation process. The user data is utilized by neural networks (NN) to train a user model and an agent/robot action policy. A decision model is used to select one output from the user model or the user simulator. The dialog agent is retrained to adapt to a specific user. The agent policy is a combination of RL-based and NN-based policies.

1) User Model: We use the Multi-Layer Perceptron (MLP) algorithm to train the user model, which is used to model the user preference and replace the user to provide feedback during the simulation-based training. The input feature S_u is the first 6 components of the state $S_u = \{S_1, S_2, S_3, S_4, S_5, S_6\}$, which is a 91-D vector. The output is a 3-D vector representing "accept", "reject" or "others". In order to model a user, the most important thing is to collect the user data from the interaction. We proposed different ways to collect user data. Firstly, when the agent initiates a conversation ($type_1$), the user feedback is recorded. This data is represented by D_f. Secondly, when the user initiates a conversation ($type_2$), the user intent is classified using NLU module. If the user intent belongs to one of the Agent Actions shown in Table I except $do\ nothing$ and there is a sound activity, the robot asks the user for confirmation. If confirmed, S_6 is replaced by the user intent and this generated S_u is recorded as "accept". This data is represented by D_i. In order to obtain more data, we assume that the agent action proactively initiated by the user is his/her favorite one. The rest of the agent actions are used to generate 13 training samples marked as "reject". This generated data is represented by D_g. For the same S_u, the latest data sample from D_f and D_i can replace the data in D_f, D_i and D_g. The user model is updated after every N interactions. As described above, the User Modeling Algorithm (UMA) is proposed as shown in Table II.

TABLE II: User Modeling Algorithm.

Input: state S_u, interaction index idx, model update interval N, conversation type tp, user data $D = \varnothing$, user response UR.
Output: user model f_μ.

Function data_collection(D, S_u, tp, UR):
  If tp == $type_1$, **Then**
    remove_duplicate(S_u,UR); $D_f = D_f \cup (S_u,UR)$;
  If tp == $type_2$, **Then**
    S_u[-1] = UR;
    remove_duplicate(S_u,"accept"); $D_i = D_i \cup (S_u$,"accept");
    new = generate_new(S_u,"reject");
    remove_duplicate(new,"reject"); $D_g = D_g \cup$ (new,"reject");
  $D = D \cup D_f \cup D_i \cup D_g$;
  Return D;
D = data_collection(D, S_u, tp, UR);
If $idx\%N$ == 0, **Then**
  f_μ = MLP(D);
  Return f_μ;

2) Policy Reshaping: RL-based methods have an ability to carry out exploration and exploitation. However, it also suffers from forgetting user feedback data. Supervised learning methods like neural networks (NN) have an ability to memorize user data. But when it comes to performing as an agent policy, the supervised learning methods suffer from over-fitting when there is not enough user data and sample inefficiency because only the "accept" user feedback can be used. Therefore, we proposed to reshape the SAC policy by combining the SAC policy with a supervised learning-based policy. We use the MLP algorithm to model the policy, which has the same dimensions of input and output as the SAC policy, and the "accept" user feedback data are used. Eq. (5) gives the final policy after Policy Reshaping (PR). β is the importance weight, which is set to be 0.7 based on our test.

$$\pi(s) = \beta\pi_\theta(s) + (1-\beta)MLP_{policy}(s) \tag{5}$$

3) Adaptation Process: Table III shows the Context-Aware Conversation Adaptation Algorithm (CACAA), which demonstrates the adaptation process. In Step 1, the dialog agent interacts with the user simulator to train an initial policy. It is used in Step 2 to interact with the user. In Step 2, or the adaptation stage, the user data D is obtained

from HRI, which is used to train a user model f_μ and a supervised learning-based policy MLP_{policy}. The Decision Model (DM) decides the acceptance of the user model based on f_μ's output. After adaptation, the adapted policy π is deployed to interact with the user and wait for the next round of adaptation.

TABLE III: Context-Aware Conversation Adaptation Algorithm.

Input: initial user preference up, user data $D = \varnothing$,train_days, epochs, probability threshold $thres$, importance weight β.
Output: adapted policy π.

#Step 1: Train an initial policy;
Randomly initialize π_θ, $Q_{\omega 1}$, $Q_{\omega 2}$;
For day **in** train_days:
  $h = \varnothing$; buffer $b = \varnothing$; $t = 0$; s_t=generate_state(t,h,up);
  For t **in** epochs:
    #Table **??**, Eq.(4)
    a_t=arg $\max_{a \in A}(\pi_\theta(s_t))$; ur=generate_response(up, s_t, a_t);
    $h = h \cup (s_t, a_t)$; s_{t+1}=generate_state(t,h,up);
    $b = b \cup (s_t, a_t, ur, s_{t+1})$;
    $Q_{\omega j} \leftarrow Q_{\omega j}$ - lr* $\frac{\partial L_{critic}(\omega_j)(b)}{\partial \omega j}$, j=1,2; #Eq.(1)
    $\pi_\theta \leftarrow \pi_\theta$ - lr* $\frac{\partial L_{actor}(\theta)(b)}{\partial \theta}$; #Eq.(3);

#Step 2: Adapt the robot policy based on user data;
Function DM(up, f_μ, s_t, a_t):
  predicted_response, prob = $f_\mu(s_t)$;
  If predicted_response != $no\ response$ & prob $> thres$, **Then**
    Return predicted_response;
  Else Return generate_response(up, s_t, a_t); #Table **??**
Randomly initialize MLP_policy;
While True
  D_{new}, S_u, tp, UR = HRI(real user, π);
  $D = D \cup D_{new}$;
  f_μ = UMA(D, S_u, tp, UR); # Table II
  If f_μ is new, **Then**
    $MLP\_policy \leftarrow MLP\_policy(D)$;
    For day **in** train_days:
      For t **in** epochs:
        $h = \varnothing$; buffer $b = \varnothing$; $t = 0$; s_t=generate_state(t,h,D);
        For t **in** epochs:
          target policy π = $\beta\pi_\theta$ + $(1-\beta)MLP_{policy}$; # Eq.(5)
          a_t=arg $\max_{a \in A}(\pi(s_t))$; ur=DM(up, f_μ, s_t, a_t);
          $h = h \cup (s_t, a_t)$; s_{t+1}=generate_state(t,h,D);
          $b = b \cup (s_t, a_t, ur, s_{t+1})$;
          $Q_{\omega j} \leftarrow Q_{\omega j}$ - lr* $\frac{\partial L_{critic}(\omega_j)(b)}{\partial \omega j}$, j=1,2; #Eq.(1)
          $\pi_\theta \leftarrow \pi_\theta$ - lr* $\frac{\partial L_{actor}(\theta)(b)}{\partial \theta}$; #Eq.(3);
  π = $\beta\pi_\theta$ + $(1-\beta)MLP_{policy}$; # Eq.(5)

V. Experimental Evaluation

A. Context Recognition Evaluation

1) Experimental Setup: To train a sound-based daily activity recognition model, for each of the 33 home environment sound events, we collected 200 audio segments. Each audio has a duration of 1 second. The audio sources include Google Audio Set [21], CHIME-HOME [22], FreeSound [23] and self-recorded audios. We used a smart phone to play the sound in our smart home testbed [5] at different locations and distances. The channel is set to mono. The sampling rate is unified to 32,000Hz. The data is divided into a training set and a testing set with a ratio of 9:1. Therefore, there are 5,940 audios for training and 660 audios for testing. To train image-based home scene recognition, we collected 200 images for each home scene in our smart home testbed. The resolution is 640 x 360. The data is also divided into a training set and a testing set with a ratio of 9:1. Therefore, there are 1,080 pictures for training and 120 audios for testing.

2) Results and Analysis: We fine-tuned our collected images on the publicly available pre-trained ResNet model for 3 epochs and obtained an accuracy of 100% on the testing set. For sound-based activity recognition, we trained the model for 200 epochs and obtained an accuracy of 94.7% on the testing set. We observed that most of sound activities can be recognized at an accuracy of 100%. Even with a good test performance, it should be noted that our testbed is an ideal environment. To use the recognition models in real environments, we need more training data to enhance the model.

B. Adaptation Evaluation on Human Subjects

1) Experimental Setup: We recruited 8 human subjects aged between 25 and 35 to test the CACAS, including 7 males and 1 female. The human subjects are all university students and 3 of them are familiar with robots. The human subject test was approved by the Oklahoma State University IRB office under application No. IRB-22-252. We introduced the system to the participants before we asked them to describe 5 preferences that they want the robot to behave to assist their daily life. Those preferences are only used for testing purposes and the system has no access to them. Both the robot and the user can initiate an action. When the robot initiates an action, the user can accept, reject or not respond to the action. If the user does not respond, the interaction is not recorded. When the user initiates an action, two interaction data points are recorded: the initiated action is regarded as accepted and the robot action "do nothing" is regarded as rejected in the current state.

The adaptation process was conducted in our ASCC Smart Home testbed [24], where the participants interacted with the robot [25]. The testbed mimics an apartment, which includes a bedroom, a living room, a kitchen, a bathroom and a dining room. The participants choose a room and the robot stays with the user. They can also use different rooms for different conversations. In order to conduct the experiment in an efficient manner, we developed a web page as a user interface to set the context. Users can use the browser on a tablet to select the sub-states S_3, S_4, and S_1 that they want to perform. The sound of the activity is played by the speaker. The rest of the sub-states are obtained automatically by the robot. Participants were asked to perform their daily routines that follow their preferences and interact with the robot around 15 times a day and last for 7 days. After each interaction day, we utilize the proposed CACAS to adapt the policy. After adaptation, the participant can interact with the robot in the next day. It took each participant around 50 minutes to finish the test. The attached video demonstrated the adaptation process of the robot using the proposed method. To evaluate users' experience with the proposed adaptation system, we

provided a post-test questionnaire which asks: "What is your satisfaction level regarding the robot's ability to adapt to your preferences?" This question uses a linear scale, with 1 meaning very dissatisfied and 10 meaning very satisfied.

For comparison purpose, we also implemented 3 other methods: a) the CACAS method without policy reshaping (CACAS w/o PR), b) the original Soft Actor-Critic reinforcement learning method (ORL) adapted using user data, and c) the MLP policy, which is a neural network-based supervised learning method (NN). We did not redo the experiment using the 3 comparison methods since it will take a long time. Therefore, we used the interaction data each day to adapt those models and tested them by generating data using the described preferences.

2) Results and Analysis: Fig. 3 shows the average of all users' success rate and the cumulative number of feedback for 7 test days. The boundaries of the shaded region show the maximum and minimum value of all users' data. A successful event is defined as the robot initiates an action and the user accepts it. Initiating an action by the user is not regarded as a successful event. Fig. 4 shows the comparison results of the 4 methods obtained from the user-described preferences. The results indicate the following:

- The average number of states covered by the 5 preference descriptions of each user is 1564 according to our calculation. From Fig. 3, we can observe that on the first day, the success rate is only 30%, which is the result of the initial model. After 7 days of adaptation, CACAS achieves a success rate of 85% with a total of 104 user interaction data. The success rate of the seventh day is obtained from the training using the first 6 days' data. Even with a large preference space, CACAS only has two or three wrong robot actions on average in a day's interaction after 7 days. The post-survey showed that 87.5% of the participants rated their satisfaction with the adaptation results at 8 or higher. The mean satisfaction rating is 8.125, with a standard deviation of 0.64.
- From Fig. 4, we can observe that on the last day, the CACAS achieves the highest success rate of 89%, which is 19%, 36% and 68% higher than CACAS w/o PR, ORL, and NN, respectively. CACAS also has the best performance in all the test days. Even without policy reshaping (CACAS w/o PR), the proposed method still outperforms the original reinforcement learning and supervised learning methods.

From the human subject test, CACAS achieves a success rate of 85% and 87.5% of participants were satisfied with the adaptation results. From the simulation test based on participants' preferences, CACAS has the highest success rate compared with the baseline methods. The proposed CACAS has a good adaptation ability and is well accepted by human subjects.

In the current setting, the robot is only given a very limited number of actions to choose from, which is the limitation of the current work. Some participants mentioned that they need other skills than those on the skill list. We need to enlarge

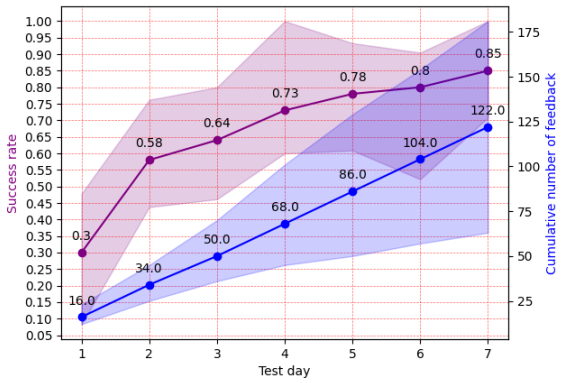

Fig. 3: Success rate and cumulative number of feedback.

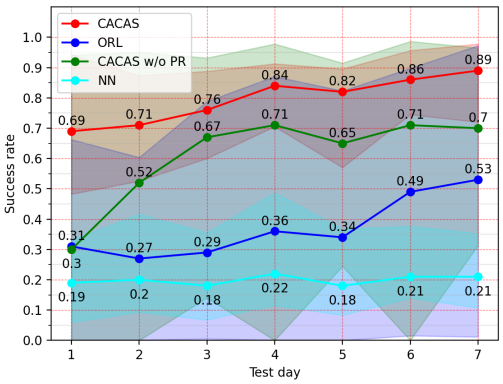

Fig. 4: Preference prediction comparison.

the robot action set and explore how to enable the robot to expand its skills automatically according to users' needs in the future. Besides, relying solely on sound to recognize activities has certain limitations in real environments. For example, the sound event may come from other rooms, other people or even pets. Therefore, it would be important to distinguish the location of the sound and the subject that generates it.

VI. Conclusion and Future Work

In this paper, we proposed a context-aware conversation adaptation system for HRI in which the robot can proactively initiate a conversation and effectively adapt to different users in particular contexts. We evaluated the CACAS with both simulated and real users. The results show that our context-aware conversation adaptation system achieves a good performance, which is practical for robot deployment. In the future, we will enhance the sound-based activity recognition model by collecting more data and test the performance in real home environments, which requires the robot to distinguish the location and source of the sound events. We will also recruit more participants to evaluate the proposed method.

REFERENCES

[1] Jibo, https://jibo.com/ accessed in September, 2022.

[2] ElliQ, https://elliq.com/ accessed in September, 2022.

[3] Amazon Astro, https://www.amazon.com/Introducing-Amazon-Astro/dp/B078NSDFSB accessed in September, 2022.

[4] Z. Su, F. Liang, H. M. Do, A. Bishop, B. Carlson, and W. Sheng, "Conversation-based medication management system for older adults using a companion robot and cloud," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2698–2705, 2021.

[5] H. M. Do, K. C. Welch, and W. Sheng, "Soham: A sound-based human activity monitoring framework for home service robots," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 3, pp. 2369–2383, 2022.

[6] S. Trick, D. Koert, J. Peters, and C. A. Rothkopf, "Multimodal uncertainty reduction for intention recognition in human-robot interaction," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 7009–7016.

[7] B. Abbasi, N. Monaikul, Z. Rysbek, B. D. Eugenio, and M. Žefran, "A multimodal human-robot interaction manager for assistive robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6756–6762.

[8] H. Wang, X. Li, and X. Zhang, "Multimodal human-robot interaction on service robot," in *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, vol. 5, 2021, pp. 2290–2295.

[9] G. De Magistris, R. Caprari, G. Castro, S. Russo, L. Iocchi, D. Nardi, and C. Napoli, "Vision-based holistic scene understanding for context-aware human-robot interaction," in *AIxIA 2021 – Advances in Artificial Intelligence*, S. Bandini, F. Gasparini, V. Mascardi, M. Palmonari, and G. Vizzari, Eds. Cham: Springer International Publishing, 2022, pp. 310–325.

[10] A. Tapus and M. Mataric, "Socially assistive robots: The link between personality, empathy, physiological signals, and task performance." in *Proceedings of the AAAI spring symposium on emotion, personality and social behavior*, 2008, p. 6.

[11] H. Ritschel, T. Baur, and E. André, "Adapting a robot's linguistic style based on socially-aware reinforcement learning," in *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2017, pp. 378–384.

[12] H. W. Park, I. Grover, S. Spaulding, L. Gomez, and C. Breazeal, "A model-free affective reinforcement learning approach to personalization of an autonomous social robot companion for early literacy education," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 687–694, Jul. 2019. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/3846

[13] K. Tsiakas, M. Abujelala, and F. Makedon, "Task engagement as personalization feedback for socially-assistive robots and cognitive training," *Technologies*, vol. 6, no. 2, 2018. [Online]. Available: https://www.mdpi.com/2227-7080/6/2/49

[14] E. Ferreira and F. Lefèvre, "Reinforcement-learning based dialogue system for human–robot interactions with socially-inspired rewards," *Computer Speech Language*, vol. 34, no. 1, pp. 256–274, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0885230815000364

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[16] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *In International Symposium on Music Information Retrieval*, 2000.

[17] E. Levin, R. Pieraccini, and W. Eckert, "Learning dialogue strategies within the markov decision process framework," in *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, 1997, pp. 72–79.

[18] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *CoRR*, vol. abs/1801.01290, 2018. [Online]. Available: http://arxiv.org/abs/1801.01290

[19] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, "Trust region policy optimization," *CoRR*, vol. abs/1502.05477, 2015. [Online]. Available: http://arxiv.org/abs/1502.05477

[20] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[21] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[22] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "Chime-home: A dataset for sound source recognition in a domestic environment," in *2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2015, pp. 1–5.

[23] FreeSound, https://freesound.org/ accessed in September, 2022.

[24] H. M. Do, M. Pham, W. Sheng, D. Yang, and M. Liu, "Rish: A robot-integrated smart home for elderly care," *Robotics and Autonomous Systems*, vol. 101, pp. 74 – 92, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889017300477

[25] H. M. Do, W. Sheng, E. E. Harrington, and A. J. Bishop, "Clinical screening interview using a social robot for geriatric care," *IEEE Transactions on Automation Science and Engineering*, pp. 1–14, 2020.