Batch Recurrent Q-Learning for Backchannel Generation Towards Engaging Agents

Nusrah Hussain, Engin Erzin, T. Metin Sezgin, and Yücel Yemez
College of Engineering, Koç University, Istanbul, Turkey
E-mail: {nhussain15, eerzin, mtsezgin, yyemez}@ku.edu.tr

***Abstract*—The ability to generate appropriate verbal and non-verbal backchannels by an agent during human-robot interaction greatly enhances the interaction experience. Backchannels are particularly important in applications like tutoring and counseling, which require constant attention and engagement of the user. We present here a method for training a robot for backchannel generation during a human-robot interaction within the reinforcement learning (RL) framework, with the goal of maintaining high engagement level. Since online learning by interaction with a human is highly time-consuming and impractical, we take advantage of the recorded human-to-human dataset and approach our problem as a batch reinforcement learning problem. The dataset is utilized as a batch data acquired by some behavior policy. We perform experiments with laughs as a backchannel and train an agent with value-based techniques. In particular, we demonstrate the effectiveness of recurrent layers in the approximate value function for this problem, that boosts the performance in partially observable environments. With off-policy policy evaluation, it is shown that the RL agents are expected to produce more engagement than an agent trained from imitation learning.**

***Index Terms*—human-robot interaction, engagement, partially observable Markov decision process, batch reinforcement learning**

I. Introduction

The development of social robots targets the design of features that may allow communication and cooperation with humans as naturally as possible. These may have diverse applications like health care, education, quality of life, entertainment and collaboration [1]. However, it can be quite challenging to measure the quality of interaction with social robots [2]. The ability to engage a user in the social task is a characteristic of user experience [3], [4]. This feature is particularly important for applications like tutoring [5], companionship [6] and instructive agents [7]. Several definitions of engagement exist in the literature, which have been described in detail by Glas et al. [8]. Poggi describes engagement as "the value that a participant in an interaction attributes to the goal of being together with the other participant(s) and of continuing the interaction" [9]. In direct face-to-face scenarios, backchannels like non-verbal gestures (nods and smiles), non-verbal vocalizations (mm, uh-huh, laughs) and verbal expressions (yes, right) have been shown to play a significant role in enhancing engagement of the user [10], [11]. In this paper, we focus on training an agent to enhance the engagement of a user by generating appropriate backchannel events during a human-robot conversation.

Generation of backchannels may be described as a sequential decision-making process during a dialog. These events are triggered based on some history of the interaction and they may impact the future course of the interaction. For example, a conversation may conclude prematurely in front of an unresponsive partner, while it may become very uncomfortable if the person in front laughs all the time. Though the objective behind backchannel generation by a human is quite complex, we chose to learn its optimization for increasing engagement of a user. Being a sequential-decision making problem, we propose to formulate it as a Markov decision process (MDP) where a policy decides an action given the current state in order to maximize some notion of reward. As in many real-world problems, the complete state information may not be available and instead, partial observations with some distribution over the true state may only be extracted. Therefore, we approach this problem as a partially observable MDP and similar to the work proposed by Hausknecht et al. [12], we use LSTM (long short-term memory) layers to embed history into the learning process. The rewards are described by the engagement measurement of the user and model-free value-based reinforcement learning (RL) methods are used as the optimization techniques to solve the partially observable MDP.

Reinforcement learning techniques are generally implemented as an online learning technique where the agent trains as it interacts with the environment. However, in applications where the agent is required to interact with a human, it will be highly time-consuming, intensely tiring and will demand much patience from the interacting person, especially when dealing with initial randomized policy. There are numerous other real-world applications that face similar problem like personalized curriculum suggestion for a student [13], online advertisement [14] and medical treatment recommendation systems [15] where a bad policy may even be dangerous and illegal. To avoid such tedious interactions with the environment, batch RL techniques allow an agent to learn from a fixed dataset without further interactions with the environment. The dataset is gathered by a more controlled policy like a human. Inspired by this key advantage of the batch RL and the availability of recorded human-to-human datasets on dyadic interactions, we

978-1-7281-3888-6/19/$31.00 ©2019 IEEE

approach our training as a batch RL problem. We work with the IEMOCAP (interactive emotional dyadic motion capture) dataset [16] and process it to define and extract tuples of the form $< s_t, a_t, r_t, s_{t+1} >$, which are required by batch RL. Here s_t is the present state, a_t is action taken, r_t is the reward received and s_{t+1} is the next state the environment transitions to. It is important to note that the IEMOCAP dataset does not represent optimal trajectories for maximizing engagement. It was designed to cover a range of emotional behaviors. Therefore, instead of learning to imitate the behavior policy, the goal is to extract the best policy from the batch data. The success of the training is determined by Bellman residuals and the engagement expected from the learned policy.

II. Background

A. Markov Decision Process

In general, the reinforcement learning formulates the optimization problem as a Markov decision process (MDP) $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$ in which at each time step t, the environment observes a state $s_t \in \mathcal{S}$, the agent takes an action $a_t \in \mathcal{A}$ and the scalar reward $r_t \sim \mathcal{R}(s_t, a_t)$ is generated by the environment. Then the environment makes a transition to the next state $s_{t+1} \sim \mathcal{P}(s_t, a_t)$. The discount factor $\gamma \in [0, 1)$ weighs the future rewards, determining the extent of temporal data that is affected by the current action. The solution to any MDP is a policy $\pi(a|s)$ which maximizes the expectation of sum of discounted rewards, i.e., the return.

In real-world environments, it is more common to receive observations that partially reflect the true underlying states. That is, the Markov property may rarely hold. Therefore, learning from a partially observable MDP (POMDP) may prove to be more beneficial since this formulation explicitly acknowledges that there is a distribution of possible states given an observation. Formally, a POMDP can be described as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{O}, \Omega, \gamma)$. Here, $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma)$ are defined as earlier and additionally observation $o \in \Omega$ is generated from the probability distribution $o \sim O(s)$.

B. Q-learning

Q-Learning [17] is a model-free off-policy algorithm for estimating the long-term expected return of executing an action from a given state. These estimated returns are known as Q-values. A higher Q-value indicates an action a is judged to yield better long-term results in a state s. Q-learning follows the iterative process of fitting the Bellman control equation [18] given by:

$$Q(s_t, a_t) = r_t + \gamma \max_{a'} Q(s_{t+1}, a') \tag{1}$$

So the loss function L is defined by the mean square of the difference between right hand side and left hand side of the Bellman control equation. Deep Q-learning models the Q-values using neural networks, and hence the Q-function is represented by parameters θ. The update equation is then given as:

$$\theta_{i+1} = \theta_i + \alpha \nabla_\theta L(\theta_i) \tag{2}$$

Vanilla Deep Q-Learning has no explicit mechanisms for deciphering the underlying state of the POMDP and is only effective if the observations are reflective of underlying system states. In the general case, estimating a Q-value from an observation can be arbitrarily bad since $Q(o; a|\theta) \neq Q(s; a|\theta)$. The deep recurrent Q-learning architecture by [12] shows that adding recurrency allows the Q-network to better estimate the underlying system state, narrowing the gap between $Q(o; a|\theta)$ and $Q(s; a|\theta)$.

C. Batch Reinforcement Learning

Reinforcement learning (RL) algorithms generally fall into the category of online algorithms. As the agent interacts with the environment, it updates its policy towards high rewarding actions. However, this also means that the agent forgets its past experiences and cannot re-utilize the data from the state regions that were visited earlier. Batch reinforcement methods encourage first the collection of data and then learning of the policy from this batch data in an offline manner. This may be repeated several times but a pure batch reinforcement learning method performs one step of data collection followed by one step of offline learning. A more detailed survey on batch reinforcement learning can be found in [19]. Experience replay is a similar concept which initializes a fixed capacity of a buffer and keeps pushing new samples into the buffer and popping the old ones. Deep Q-network (DQN) [20] has shown that sampling randomly from the buffer to make updates, and hence breaking the sequential correlation between the samples, improves performance on Atari games. Another advantage of batch RL, that is of more interest to us, is that the batch data may be collected by any behavior policy, that may even be random. Several works exist on algorithms for batch reinforcement learning. The fitted Q-iterations (FQI) [21] and neural fitted Q-learning (NFQ) [22] are among the more popular algorithms. The former uses a tree-based approach to model a Q-network while later modeled the Q-network with multi-layer perceptrons and fitting the Bellman optimality equation.

III. Recent Work

A. Social Robots and RL

Reinforcement learning has shown much success in a variety of domains and is a trending technique in the field of robotics. Several works have shown its use in training of an agent for behaviors similar to that of humans. The works by Qureshi et al. [23] [24] presented an RL method for training an agent to greet as humans with the sequential actions of wait, look, wave and shake hand. They used multi-modal DQN and generated rewards at every successful handshake. The robot was trained for 14 days while it interacted with humans. In the work by Mitsunaga et al. [25], RL is employed to adjust motion speed, timing, interaction distances, and gaze in the context of human-robot interaction (HRI). The reward is based on the amount of movement of the subject and the time spent gazing at the robot in one interaction. Recurrent neural networks were used in combination with Q-learning by Lathuilière et al. [26] to find an optimal policy for robot gaze control in HRI. The training was performed in a simulated environment. In all these works, however, the agent either interacts with the environment (humans) for several days or training is done using simulators. We address the challenge where experience on a real physical system may be tedious to obtain, expensive, time-consuming and hard to simulate. We propose to use human-to-human interaction datasets as a batch of off-policy samples (trajectories) and use them in the context of offline batch reinforcement learning.

B. *Engagement in Interactions*

Poggi [9] describes engagement as "the value that a participant in an interaction attributes to the goal of being together with the other participant(s) and of continuing the interaction". An agent trained to maintain the engagement of a user is vital for several applications like companionship, tutoring, and ambient assisting living. A survey by Clavel et al. summarizes the issues regarding engagement in human-agent interactions, emphasizing its importance and indicating the growing interest of researchers in the field [27]. Verbal and non-verbal backchannels like nods, head tilts, eye-gaze, hmms etc. are an important aspect of engagement and have been shown to promote engagement and interest levels of the user [10], [11]. Researchers have mainly focused on rule-based backchannel generation [28], [29] or data-driven unsupervised methods [30]. In this work, we show how to formulate the problem in a reinforcement learning framework and train an agent to learn a policy for backchannel generation that maximizes the engagement of the user.

One of the pioneering studies on the measurement of engagement is the work by Rich et al. [31], where the authors propose an engagement model for collaborative interactions between human and computer. They define four types of events as engagement indicators, referred to as connection events (CEs), which include directed gaze, mutual facial gaze, adjacency pair, and backchannels. Directed gaze event is defined when both participants look at a nearby object related to the interaction at the same time. The mutual facial gaze occurs when there is face-to-face eye contact. Adjacency pair indicates a successful event when turn taking occurs with some minimal time gap. Finally, backchannels refer to the generation of audio-visual feedback by a listener during the speaker's turn. In our work, we use these connection events to quantify engagement and generate a single scalar value at each time step to represent the rewards. An alternative option may be to directly annotate the engagement levels in the dataset. However, automatic detection of engagement allows the refinement of the policy in the future by continuously updating the policy as the agent interacts with humans.

IV. Proposed Methodology

As described earlier, we propose a method to train an agent for the generation of backchannels that may maximize engagement using datasets as batch data. We work with the IEMOCAP (interactive emotional dyadic motion capture) dataset [16] that consists of dyadic human-to-human conversations on a range of scenarios. A total of 151 dialogues were performed by 10 professional actors in pairs on scripted and improvised scenes. In order to treat this as a batch data, we assume that of the two actors, one represents a behavior policy which takes the actions and the second actor behaves as an environment that generates states and rewards. Thus the IEMOCAP dataset may be viewed as a batch of trajectories collected by the behavioral policy. In order to double the training data, we also switch the roles of the actor as the behavior policy and the environment. Thus in total, we have 302 sequential decision making trajectories.

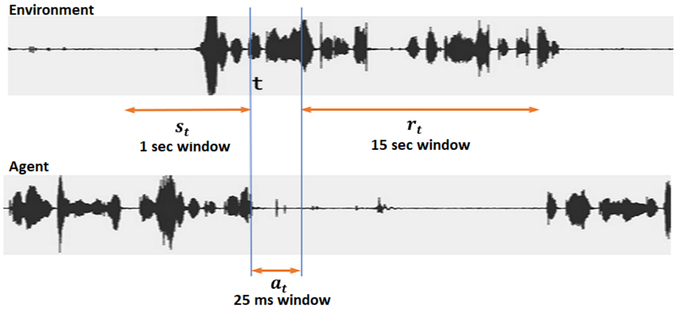

Fig. 1. Reinforcement learning formulation of speech driven backchannel generation (not drawn to scale)

A. *Batch-RL Formulaion*

Batch reinforcement learning algorithms work with tuples of the form $\langle s_t, a_t, r_t, s_{t+1} \rangle$ for $t = 1 : T$. At time t, s_t is the state of the environment, a_t is the action taken by the agent and r_t is the reward. We extract these tuples at a rate of 40 Hz from the dataset, hence a batch data of approximately 3 million tuples is produced. But the first step is to lay down the definitions of states, actions and rewards. Though our framework is general for any backchannel event, we perform experiments with laughs as a backchannel that may enhance engagement. States, actions, and rewards are defined as follows:

- **State:** The state of the environment is represented by speech features extracted from past one second of data at every 25 msec step. This produces state information at a rate of 40 Hz.
- **Action:** Agent's action is a binary variable, indicating the absence or presence of laugh. Laughs of the actor described as the behavioral policy are labeled at a rate of 40 Hz.
- **Reward:** The reward is a scalar quantity which comes from the engagement measures of the user at every time step. Engagement is calculated by determining the number of connection events in a time window. It is further elaborated below.

The states are defined using the mel-frequency cepstrum coefficients (MFCCs) and prosody features extracted from the speech signal of the environment. 13-dimensional MFCC features are computed using 40 milliseconds sliding Hamming window at intervals of 25 milliseconds. The speech intensity, pitch, and confidence-to-pitch with their first derivates make up a 6-dimensional prosody feature, so a 19-dimensional feature vector is formed when MFCCs and prosody are concatenated as in [32]. Following this, feature summarization is performed where a set of statistical quantities are computed that describe the short-term distribution of each feature over the past one second. These quantities comprise eleven functions, more specifically mean, standard deviation, skewness, kurtosis, range, minimum, maximum, first quantile, third quantile, median quantile and inter-quartile range, which were successfully used before by [33]. The dimension of each of these statistical feature vectors is 11 times the dimension of the corresponding feature vector. This makes the feature size of length 209. Fig. 1 shows the time windows used to extract states, rewards, and actions in one tuple. The dataset is pre-processed and such tuples are saved in a buffer.

Our measure of engagement is based on the method proposed in [31], which is applicable to face-to-face collaborative HCI scenarios. Similar to the description in Section III-B, we use the connection events (CE) (1) mutual facial gaze, (2) adjacency pair and (3) backchannels (that include laughs, smiles, nods and head-shakes) to quantify engagement. In [31], the directed gaze event is defined when the agent and the participant look at a nearby object related to the interaction at the same time. However, in our dataset, since we do not have objects of interest at which both parties look at, we exclude it in our definition. The extracted CEs are then used to calculate a summarizing engagement metric called mean time between connection events (MTBCE). MTBCE measures the frequency of successful connection events that is for a given time interval T, MTBCE is calculated by T / (no. of CEs in T). As MTBCE is inversely proportional to engagement, similarly to [31], we use pace = 1/MTBCE to quantify the engagement between a participant and the robot. The pace measure is calculated over a window of 15 seconds in our experiments.

B. Q Networks

We perform experiments with two different neural networks and evaluate their relative performance. The first function approximation network is modeled as a multi-layer perceptron (MLP) to solve for a MDP where states follow the Markov property. It consists of 209 inputs (state feature size), two hidden layers with 100 and 25 neurons respectively and 2 outputs (Q-values for the two actions). For all the neurons ReLU activation function is used. For the second neural network, we introduce a LSTM layer to handle the problem as a POMDP. The same MLP structure is used. Only its first fully connected (FC) layer is replaced with a LSTM with identical number of neurons (i.e. 100). We also experimented with an LSTM replacing the second FC layer instead. However, since it did perform as well, we include here only the results for the case where the first FC layer is replaced.

V. Experiments

For all our experiments the batch data is split into train and test sets in the ratio 4:1, hence 5-fold training is performed. It is split as leave one subject out (LOSO) and hence the reported results are subject independent. In all the experiments, the optimization is performed using Adam optimizer and a discount factor of value 0.99 is used. We train the two networks described in Section IV-B: multi-layer perceptron (MLP) and fully connected LSTM (FC-LSTM). Following the concept of experience replay and randomization proposed by DQN [20], the batch data is shuffled prior to training of the MLP network. In our second experiment, the introduction of LSTM means the tuples cannot be randomized because the network now requires sequential data input. However, if each dialog is passed sequentially, the LSTM faces stability problem due to the long lengths of the sequences and the randomization proposed by DQN cannot be incorporated. We approach this by selecting randomly a starting position in each dialog and choose only the next L time steps to pass to the network. We have tested with different sequence lengths L and found $L = 80$ (2 seconds of data) to be stable while improving the results.

We have also tested with a linear function approximation network and found it to be unstable with diverging errors. Another comparison we have performed is with a policy learned from supervised learning. For supervised learning, we use the identical MLP network with a softmax layer at the end to produce probabilities of laugh and no laugh events. The loss is defined by the cross-entropy loss function with laugh labels as the true outputs. The value estimated by this policy is used as a baseline.

VI. Results

Evaluation of the resultant policy is a challenging problem since the environment (i.e., the human participant) is not readily available in our case. Although it is possible to conduct experiments with human-robot interactions, it is desirable to first understand the policy's effectiveness using quantitative measures. We use the Bellman residual and off-policy evaluation (OPE) techniques to understand the effectiveness of each policy

A. Bellman Residual

For a Q-value function approximation network Q_θ, the Bellman residual is defined as the difference between the two sides of a Bellman control equation [34]. A smaller residual error means that the learned policy is closer to the optimal policy and is a true Q-function since it follows the Bellman equation more closely. Similar to the work of [21], we compute the Bellman residual, B_r, over the entire batch data $\mathcal{B}$ as

$$B_r = \frac{1}{|\mathcal{B}|}\sum_{\mathcal{B}}(Q_\theta(s_t, a_t) - [r_t + \gamma * \max_{a \in A} Q_\theta(s_{t+1}, a)])^2. \quad (3)$$

Fig. 2 shows the training curves for the multi-layer perceptron (MLP) and fully connected LSTM networks. Both models converge to a stable point. However, the lower Bellman

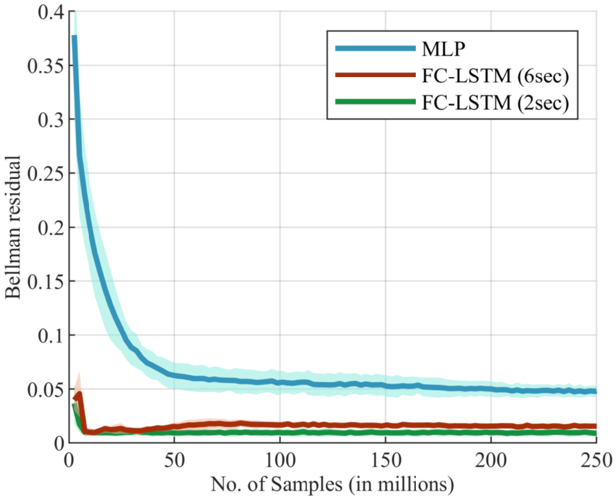

Fig. 2. Bellman residual vs training samples

residuals from the LSTM model show better optimality and validity of its value function approximation network. For comparison purpose, we also present the effect of different lengths of sequences used for LSTM training. Though truncation of sequences to L samples may result in information loss from history, longer sequences become harder to train with LSTMs. The training curves for a length of past 2 seconds of data versus past 6 seconds of data are presented here. Fig. 3 shows the trend of Q-values estimated by each network, averaged for the states present in the batch data. We observe that the estimates are close to each other, with LSTM network with 2 seconds of history surpassing marginally.

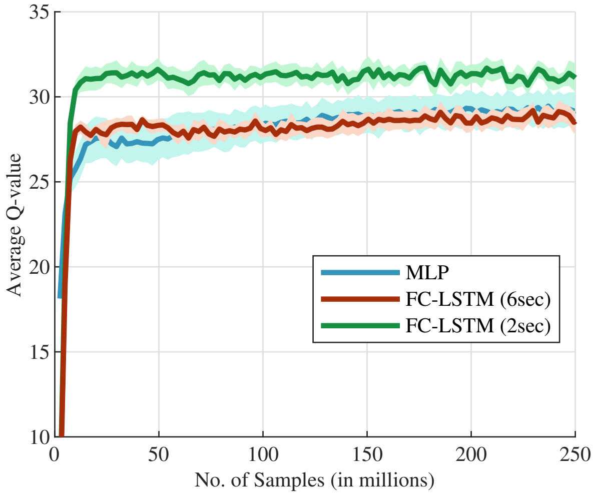

Fig. 3. Estimated Q-value vs training samples

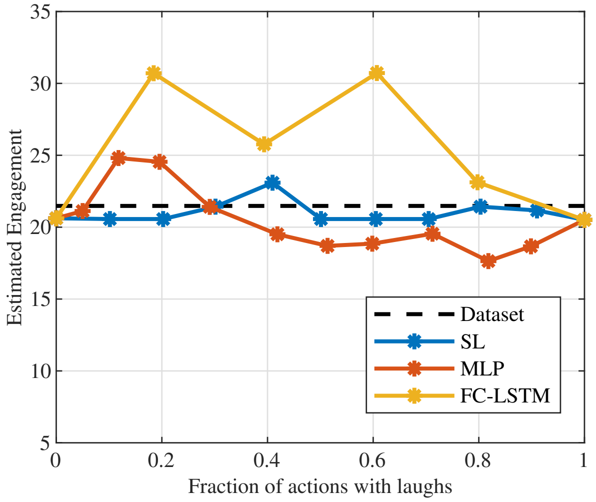

Fig. 4. Off-Policy Policy Evaluation

B. *Off-policy Policy Evaluation (OPE)*

Off-policy policy evaluation (OPE) is used to predict the performance of policy with data only sampled by a behavior policy [18]. In the last few years, many OPE techniques have emerged because of its importance in cases where a new policy cannot be tested directly with the environment [35], [36]. To compare the values of policies returned by each technique, we apply the step-wise weighted importance sampling (step-WIS) estimator given as

$$\hat{V}^{\pi}_{step-WIS} = \sum_{i=1}^{n}\sum_{t=0}^{T-1}\gamma^{t}\frac{\rho_t^{(i)}}{\sum_{i=1}^{n}\rho_t^{(i)}}r_t^{(i)}, \tag{4}$$

where n is the number of trajectories, T is the length of each trajectory and γ is the discount factor. Then, the importance weight ρ is defined as the ratio of the probability of the first $t+1$ steps of a trajectory under π to the probability under a behavior policy π_b and is given as $\rho_t = \prod_{i=0}^{t}\frac{\pi(a_i|s_i)}{\pi_b(a_i|s_i)}$. The importance sampling approach to evaluation relies on using the importance weights ρ_t to adjust for the difference between the probability of a trajectory under the behaviour policy π_b and the probability under the evaluation policy π. Following the discussion of the work in [37], the behavior policy π_b is estimated using approximate nearest neighbor [38]. Ideally OPE needs to be computed over infinite lengths, but taking into account the numerical limitations, we calculate them over trajectories of length 200 samples with shifts at every sample.

The training process for each technique produces an optimal Q-value function, which is used to implicitly define a policy. The simplest method is to act greedily and produce a deterministic policy by selecting the action with the highest Q-value. But since the OPE technique uses importance sampling and requires probabilities to reweigh each trajectory, we assign a high probability of 95% to the action suggested by the greedy policy. Also, it is observed that each agent produces laughs much more frequently than the non-laughing event. On the contrary, in the dataset the laugh events occur only $\sim 1.5\%$. In order to control the number of laughs, we use the softmax function to convert Q-values to probabilities and apply different thresholds to define a new deterministic policy. The OPE is performed at different threshold values (or amount of generated laughs) and similar to before a 95% probability is assigned to the suggested action. Fig. 4 shows the values estimated by WIS at different fraction of laughs. The fully connected LSTM clearly outperforms MLP, producing a maximum value of 30.7 versus 24.8, whereas the batch data consists of an average value of 21.47. The policy from supervised learning is also tested in a similar fashion. Different thresholds are used to limit the number of laughs and OPE is performed from the deterministic actions suggested by the policy.

VII. Conclusion and Future Work

This work has explored the effectiveness of recurrent layers for teaching non-verbal communication to robots. We have formulated the problem in a reinforcement learning framework so as to explicitly maximize engagement. More specifically, the batch Q-learning method presented in this paper aims to find an optimal policy from recorded human-to-human interactions, that generates backchannel events for an engaging interaction. As discussed in Section I, engagement is an important feature that may be used as a measure for the quality of interaction. It has been shown in various studies that the correct generation of backchannels during an interaction engages a user more effectively towards the task. In this work, we measure the engagement of an interaction by using cues like eye contact, non-verbal gestures and turn taking frequency [31]. Our experiments have focused on generating laughs as the backchannel event, but this work may easily be extended to other backchannel events such as nods and smiles. One primary goal of this work is to show that the knowledge of true state in a real-world problem may not be available and needs to be formulated as a partially observable Markov decision process. To this end, we have demonstrated the effectiveness of recurrent Q-learning by using measures such as Bellman residual, off-policy policy evaluation and mean Q-values. As future work, we will evaluate the effectiveness of our method with experiments in a human-robot interaction setting, where engagement will be monitored in real-time and backchannel events will be generated with the optimal policy learned from batch data. Furthermore, our future research direction will involve experiments with richer definitions of state incorporating features like visual landmarks, bodily gestures, emotional content and word embedding.

Acknowledgment

This work is funded by Turkish Scientific and Technical Research Council (TUBITAK) under grant number 217E040 and the first author is supported by Higher Education Commission (HEC) Pakistan.

References

[1] C. Breazeal, A. Takanishi, and T. Kobayashi, “Social robots that interact with people,” *Springer handbook of robotics*, pp. 1349–1369, 2008.

[2] K. U. R. Laghari and K. Connelly, “Toward total quality of experience: A qoe model in a communication ecosystem,” *IEEE Communications Magazine*, vol. 50, no. 4, pp. 58–65, 2012.

[3] H. L. O’Brien and E. G. Toms, “What is user engagement? a conceptual framework for defining user engagement with technology,” *Journal of the American society for Information Science and Technology*, vol. 59, no. 6, pp. 938–955, 2008.

[4] S. M. Anzalone, S. Boucenna, S. Ivaldi, and M. Chetouani, “Evaluating the engagement with social robots,” *International Journal of Social Robotics*, vol. 7, no. 4, pp. 465–478, 2015.

[5] S. D’mello and A. Graesser, “Autotutor and affective autotutor: Learning by talking with cognitively and emotionally intelligent computers that talk back,” *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 2, no. 4, p. 23, 2012.

[6] D. Griol, J. M. Molina, and Z. Callejas, “Modeling the user state for context-aware spoken interaction in ambient assisted living,” *Applied intelligence*, vol. 40, no. 4, pp. 749–771, 2014.

[7] A. Cafaro, H. H. Vilhjálmsson, and T. Bickmore, “First impressions in human–agent virtual encounters,” *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 23, no. 4, p. 24, 2016.

[8] N. Glas and C. Pelachaud, “Definitions of engagement in human-agent interaction,” in *2015 International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, 2015, pp. 944–949.

[9] I. Poggi, *Mind, hands, face and body: a goal and belief view of multimodal communication*. Weidler, 2007.

[10] B. B. Türker, Z. Buçinca, E. Erzin, Y. Yemez, and M. Sezgin, “Analysis of engagement and user experience with a laughter responsive social robot,” in *Proc. 18th Annu. Conf. Int. Speech Commun. Assoc*, 2017, pp. 844–848.

[11] B. Inden, Z. Malisz, P. Wagner, and I. Wachsmuth, “Timing and entrainment of multimodal backchanneling behavior for an embodied conversational agent,” in *Proceedings of the 15th ACM on International conference on multimodal interaction*. ACM, 2013, pp. 181–188.

[12] M. Hausknecht and P. Stone, “Deep recurrent q-learning for partially observable mdps,” in *2015 AAAI Fall Symposium Series*, 2015.

[13] T. Mandel, Y.-E. Liu, S. Levine, E. Brunskill, and Z. Popovic, “Offline policy evaluation across representations with applications to educational games,” in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1077–1084.

[14] G. Theocharous, P. S. Thomas, and M. Ghavamzadeh, “Personalized ad recommendation systems for life-time value optimization with guarantees,” in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[15] A. Guez, R. D. Vincent, M. Avoli, and J. Pineau, “Adaptive treatment of epilepsy via batch-mode reinforcement learning.” in *AAAI*, 2008, pp. 1671–1678.

[16] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J. N. Chang, S. Lee, and S. S. Narayanan, “Iemocap: Interactive emotional dyadic motion capture database,” *Language resources and evaluation*, vol. 42, no. 4, p. 335, 2008.

[17] C. J. Watkins and P. Dayan, “Q-learning,” *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[18] R. S. Sutton and A. G. Barto, *Introduction to reinforcement learning*. MIT press Cambridge, 1998, vol. 135.

[19] S. Lange, T. Gabel, and M. Riedmiller, “Batch reinforcement learning,” in *Reinforcement learning*. Springer, 2012, pp. 45–73.

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

[21] D. Ernst, P. Geurts, and L. Wehenkel, “Tree-based batch mode reinforcement learning,” *Journal of Machine Learning Research*, vol. 6, no. Apr, pp. 503–556, 2005.

[22] M. Riedmiller, “Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method,” in *European Conference on Machine Learning*. Springer, 2005, pp. 317–328.

[23] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa, and H. Ishiguro, “Robot gains social intelligence through multimodal deep reinforcement learn-

ing," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016, pp. 745–751.

[24] ——, "Show, attend and interact: Perceivable human-robot social interaction through neural attention q-network," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1639–1645.

[25] N. Mitsunaga, C. Smith, T. Kanda, H. Ishiguro, and N. Hagita, "Robot behavior adaptation for human-robot interaction based on policy gradient reinforcement learning," *Journal of the Robotics Society of Japan*, vol. 24, no. 7, pp. 820–829, 2006.

[26] S. Lathuilière, B. Massé, P. Mesejo, and R. Horaud, "Neural network based reinforcement learning for audio–visual gaze control in human–robot interaction," *Pattern Recognition Letters*, vol. 118, pp. 61–71, 2019.

[27] C. Clavel, A. Cafaro, S. Campano, and C. Pelachaud, "Fostering user engagement in face-to-face human-agent interactions: a survey," in *Toward Robotic Socially Believable Behaving Systems-Volume II*. Springer, 2016, pp. 93–120.

[28] S. Al Moubayed, M. Baklouti, M. Chetouani, T. Dutoit, A. Mahdhaoui, J.-C. Martin, S. Ondas, C. Pelachaud, J. Urbain, and M. Yilmaz, "Generating robot/agent backchannels during a storytelling experiment," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 3749–3754.

[29] C. Liu, C. T. Ishi, H. Ishiguro, and N. Hagita, "Generation of nodding, head tilting and eye gazing for human-robot dialogue interaction," in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, 2012, pp. 285–292.

[30] H. Admoni and B. Scassellati, "Data-driven model of nonverbal behavior for socially assistive human-robot interactions," in *Proceedings of the 16th international conference on multimodal interaction*. ACM, 2014, pp. 196–199.

[31] C. Rich, B. Ponsler, A. Holroyd, and C. L. Sidner, "Recognizing engagement in human-robot interaction," in *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*. IEEE, 2010, pp. 375–382.

[32] E. Bozkurt, E. Erzin, and Y. Yemez, "Multimodal analysis of speech and arm motion for prosody-driven synthesis of beat gestures," *Speech Communication*, vol. 85, pp. 29–42, December 2016.

[33] A. Metallinou, A. Katsamanis, and S. Narayanan, "Tracking continuous emotional trends of participants during affective dyadic interactions using body language and speech information," *Image and Vision Computing*, vol. 31, no. 2, pp. 137–152, 2013.

[34] L. Baird, "Residual algorithms: Reinforcement learning with function approximation," in *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 30–37.

[35] P. Thomas and E. Brunskill, "Data-efficient off-policy policy evaluation for reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 2139–2148.

[36] S. Doroudi, P. S. Thomas, and E. Brunskill, "Importance sampling for fair policy selection." *Grantee Submission*, 2017.

[37] A. Raghu, O. Gottesman, Y. Liu, M. Komorowski, A. Faisal, F. Doshi-Velez, and E. Brunskill, "Behaviour policy estimation in off-policy policy evaluation: Calibration matters," *arXiv preprint arXiv:1807.01066*, 2018.

[38] V. Hyvönen, T. Pitkänen, S. Tasoulis, E. Jääsaari, R. Tuomainen, L. Wang, J. Corander, and T. Roos, "Fast nearest neighbor search through sparse random projections and voting," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 881–888.