Nonverbal social behavior generation for social robots using end-to-end learning

Woo-Ri Ko, Minsu Jang, Jaeyeon Lee and Jaehong Kim

Abstract
Social robots facilitate improved human–robot interactions through nonverbal behaviors such as handshakes or hugs. However, the traditional methods, which rely on pre-coded motions, are predictable and can detract from the perception of robots as interactive agents. To address this issue, we have introduced a Seq2Seq-based neural network model that learns social behaviors from human–human interactions in an end-to-end manner. To mitigate the risk of invalid pose sequences during long-term behavior generation, we incorporated a generative adversarial network (GAN). This proposed method was tested using the humanoid robot, Pepper, in a simulated environment. Given the challenges in assessing the success of social behavior generation, we devised novel metrics to quantify the discrepancy between the generated and ground-truth behaviors. Our analysis reveals the impact of different networks on behavior generation performance and compares the efficacy of learning multiple behaviors versus a single behavior. We anticipate that our method will find application in various sectors, including home service, guide, delivery, educational, and virtual robots, thereby enhancing user interaction and enjoyment.

Keywords
Social robot, human–robot interaction, social behavior generation, end-to-end learning

Introduction
To optimize human–robot interactions and make them more enjoyable, social robots must possess an understanding of user behavior and the capacity to generate human-like responses (Dindo and Schillaci, 2010; Mitsunaga et al., 2008; Salem et al., 2013; Wada and Shibata, 2007). This might include behavior such as, robots greeting users upon their return home, offering a high five when a user raises a hand, or giving a hug when the user is upset. These correspond to nonverbal social behaviors that include gaze direction, hand and head gestures, torso positions, and facial expressions (Makatchev et al., 2012). To realize these nonverbal social behaviors, numerous studies have explored behavior generation methods using human behavior models or pre-defined robot motions. Breazeal and Scassellati (1999) created a software architecture capitalizing on natural human social inclinations, enabling Kismet, a facial robot, to engage in infant-like interactions with human caregivers. Furthermore, Huang and Mutlu (2012) put forth a framework founded on social science specifications of human behavior, to study the generation of social behaviors in human-like robots. Likewise, Salem et al. (2012) suggested a control architecture that empowers a humanoid robot, Asimo, to dynamically generate gestures and synchronize speech. Furthermore, Zaraki et al. (2018) went on to develop an interactive sense-think-act architecture, controlling the humanoid robot Kaspar’s behavior semi-autonomously.

Nonetheless, implementing human behavioral models or predefined robot motions requires considerable expert knowledge and human labor, which can often prove costly and time-intensive. To overcome this challenge, human-in-the-loop approaches (Wu et al., 2022) have been employed in various tasks, including manipulation planning (Raessa et al., 2020), human–robot collaboration (Weitschat and Aschemann, 2018), and reinforcement learning (Arakawa et al., 2018; Singi et al., 2023). These studies aim to create precise predictive models at a reduced cost by leveraging human knowledge and experience. Recently, with large datasets becoming publicly available (Singh and Vishwakarma, 2019; Srivastava et al., 2022; Tuyen et al.,...
data-driven learning techniques have gained traction. For example, Rahmatizadeh et al. (2018) introduced a recurrent neural network-based architecture for learning multiple manipulation tasks from demonstrations. Ahn et al. (2018) proposed a generative model that enables robots to perform diverse actions in response to input language descriptions of human behavior. Moreover, Yoon et al. (2019) devised an end-to-end learning methodology that allows robots to learn co-speech gestures of a humanoid robot through TED Talks. Similarly, Jonell et al. (2019) presented a probabilistic, generative deep-learning architecture enabling robots to learn nonverbal behaviors from YouTube videos. Expanding on this, Prasad et al. (2021) established a framework to learn handshaking behaviors solely from third-person human–human interaction data.

Although these studies have provided significant contributions to social robot behavior generation, many of these studies predominantly focus on manipulation or navigation tasks, or co-speech gestures. Several investigations have tackled nonverbal social behavior generation in robots, however, most targeted the learning of a singular social behavior (Prasad et al., 2021) or produced invalid pose sequences during long-term behavior generation (Ko et al., 2020). In this paper, we propose a Seq2Seq-based (Sutskever et al., 2014) neural network architecture capable of learning multiple nonverbal social behaviors. Predominantly used for machine translation tasks (Cho et al., 2014; Pham et al., 2019), we have repurposed Seq2Seq models to generate nonverbal behaviors in social robots. To counteract the generation of invalid pose sequences during long-term behavior (Buckchash and Raman, 2020), we have incorporated terms into the loss functions based on a generative adversarial network (GAN) (Goodfellow et al., 2014).

We trained our neural networks end-to-end using the AIR-Act2Act human–human interaction dataset introduced by Ko et al. (2021). We extracted user poses from the dataset and normalized them using a vector normalization method (Hua et al., 2019), and extracted robot poses and transformed into joint angles. To validate and quantitatively evaluate the proposed method, we performed experiments using the humanoid robot, Pepper, in a simulated environment. We also introduced novel metrics to calculate the discrepancy between the generated and ground-truth behaviors. These metrics reveal the influence of different network architectural choices on behavior generation performance and provide a comparative analysis of learning multiple behaviors versus a single behavior. The novelty of our study is three-fold.

- First, our proposed behavior generation method employs a data-driven learning technique that is capable of learning multiple nonverbal social behaviors in a complete end-to-end manner. To the best of our knowledge, this is the first attempt at learning multiple nonverbal behaviors of social robots using human–human interaction datasets such as Ko et al. (2021), along with our prior research presented in Ko et al. (2020).

- Moreover, unlike the previous study (Ko et al., 2020), GAN-based terms are incorporated into the loss functions to prevent the generation of undesirable pose sequences during long-term behaviors. Particularly, we utilize the future robot behavior as the discriminator’s input to allow the discriminator to better discriminate between desirable and undesirable behaviors, helping the model learn to generate the desirable behavior.

- Finally, given the inherent difficulty in determining the success or failure of social behavior generation compared to manipulation or navigation tasks, we also propose two additional metrics to assess the appropriateness of the generated behavior. Similar to the evaluation mechanism of humans, the proposed metrics determine whether the key or final pose of the generated behavior is generated accurately and in a timely manner.

Problem definition and assumptions

In this study, we focused on robots that interact socially with users in home environments. The task of generating robot social behavior entails deciding on the next robot behavior, $\overrightarrow{R}_{t}$, to react to the present user behavior, $\overrightarrow{U}_{t}$, while ensuring continuity with the current robot behavior, $\overrightarrow{R}_{t-1}$, at time step $t$. As portrayed in Figure 1, each behavior of the user and robot can be represented as a sequence of poses in the following format:

$$
\text{Inputs : } \overrightarrow{U}_{t} = [u_{t-m+1}, u_{t-m+2}, \ldots, u_{t}] ,
\overrightarrow{R}_{t} = [r_{t}] ,
\text{Outputs : } \overrightarrow{R}_{t} = [r_{t+1}, r_{t+2}, \ldots, r_{t+n}] ,
$$

where $u_{t}$ and $r_{t}$ represent the poses of the user and robot at time step $t$, respectively, while $m$ and $n$ are predefined numbers of poses forming $\overrightarrow{U}_{t}$ and $\overrightarrow{R}_{t}$, respectively. The user pose, $u_{t}$, can be characterized by feature points (Rapantzikos et al., 2009), a 2D or 3D skeleton model (Redmon and Farhadi, 2017; Shotton et al., 2011), a depth map (Wang et al., 2015), or an RGB image (Karpathy et al., 2014). For the robot’s movement control, the robot pose $r_{t}$ can be represented in the same way as the user pose, as joint angles (Yang et al., 2016), or as motor commands (Levine et al., 2018).

In the proposed method, we made several assumptions: (1) The user initiates the interactions and the robot only responds to them (i.e., the robot does not generate proactive behaviors). (2) The robot moves to a position where it can view the user’s hand to recognize the user’s behavior; if the user’s behavior is not recognized, the robot does not generate any behavior. (3) The robot has a 3D camera and each user pose is represented as a 3D skeleton model of the upper body. The user’s lower body is not considered as social behavior. Learning primarily involves understanding movements of the upper body. (4) The robot is a humanoid and each robot pose is represented by the joint angles of its upper body. The lower body of the robot is not considered owing to the challenges associated with maintaining the robot’s balance.
Proposed method

This section describes the proposed method for generating robot social behavior in detail. The overall procedure is depicted in Figure 2 and discussed in the Overview section. The extraction of training data is described in the Training data extraction, User pose normalization, and Robot pose transformation sections. The architecture, training process, and parameter settings for the neural network are discussed in the Neural network architecture and Training sections.

Overview

Building upon the Seq2Seq model (Sutskever et al., 2014) and GANs (Goodfellow et al., 2014), we propose a neural network architecture comprising an encoder, decoder, and discriminator. The encoder encodes the user behavior \( U_t \) into a vector \( z \). The decoder then generates the subsequent robot behavior \( R_t \), corresponding to the current robot behaviors \( R_t \) and \( z \). For real-time interaction, the next robot behavior \( R_t \) is generated at every \( n \) time step. The decoder also creates the future robot behavior \( R_{t+l} \), which serves as input for the discriminator. The discriminator determines whether a sequence of robot poses is derived from the training dataset or generated by the decoder. The ground-truth inputs and outputs utilized for training the encoder, decoder, and discriminator are extracted from human–human interaction data.

Training data extraction

We first downsampled the pose data from human–human interactions to a frequency of 10 Hz. For instance, if the initial pose data sequence is \{\( t_1, t_2, t_3, \ldots \)\} with a frequency of 30 Hz, the downsampled pose data would be \{\( t_1, t_4, t_7, \ldots \)\}. Subsequently, the pose data of the person who initiated the interaction were normalized to create a user pose \( u \), and the pose data of the person responding to the initiator were transformed into a robot pose \( r \). Further details are provided in the User pose normalization and Robot pose transformation sections. Finally, we generated the training inputs and outputs by accumulating \( m \) and \( n \) user and robot poses, where \( m \) and \( n \) were set to 15 and 5, respectively. If the user and robot pose data are denoted as \{\( u_{t_1}, u_{t_2}, \ldots \), \( r_{t_1}, r_{t_2}, \ldots \)\}, respectively, the first training input and output will be \{\( u_{t_1}, u_{t_2}, \ldots , u_{t_{15}}, r_{t_1} \)\} and \{\( r_{t_{16}}, r_{t_{17}}, \ldots , r_{t_{20}} \)\}, and the next training input and output will be \{\( u_{t_2}, u_{t_3}, \ldots , u_{t_{16}} , r_{t_1} \)\} and \{\( r_{t_{17}}, r_{t_{18}} , \ldots , r_{t_{21}} \)\}, respectively. As a result, the training input and output sequences have a length of 1.5 s and 0.5 s, respectively.

The values of \( m \) and \( n \) need to be determined empirically based on careful examination of the training dataset. The value of \( m \) should be large enough for the robot to understand the user’s intent. However, setting it too large slows model convergence due to excessive input information. Similarly, \( n \) must be sufficiently large for the robot to learn the required behavior. If it is too large, the robot will generate its next behavior solely based on the user’s very previous behavior, which can degrade learning performance.

User pose normalization

As illustrated in Figure 3, a human pose can be represented as

\[
P = [p_1, p_2, \ldots , p_9]^T,
\]

(2)
Figure 2. Our proposed neural network architecture for generating robot social behavior consists of encoder, decoder, and discriminator. encoder encodes current user behavior, decoder generates next robot behavior based on current user and robot behaviors, and discriminator determines whether a sequence of robot poses is from the training dataset or generated by decoder. Ground-truth inputs and outputs for training encoder, decoder, and discriminator are extracted from human–human interaction data.

Figure 3. Nine body joints selected to represent a user pose (i.e., torso, spine shoulder, head, shoulders, elbows, and wrists) and illustration of normalization of the user pose.
where \( p_j = [x_j, y_j, z_j] \) refers to the 3D coordinates of the \( j \)th body joint with respect to the camera’s position. In this study, we selected nine body joints (i.e., torso, spine, shoulder, head, shoulders, elbows, and wrists) to represent the user’s pose, as demonstrated in Figure 3. To improve the stability and performance of the neural network model, and to emphasize the active movement of the body joints, we adopted the vector normalization method proposed in Hua et al. (2019). This normalization can be defined as follows:

\[
\mathbf{u} = \left[ v_1^1, v_2^2, v_3^3, v_4^4, v_5^5, v_6^6, v_7^7, v_8^8, d \right]^T,
\]

where \( v_j = (p_j - p_i)/ \|p_j - p_i\| \) represents the normalized direction vector from the \( i \)th body joint to the \( j \)th body joint. The value of \( d = \|p_1\|/d_{\text{max}} \) is the normalized distance from the camera to the torso, and \( d_{\text{max}} = 5 \text{ m} \) is the maximum distance observed in the dataset. Consequently, the size of the normalized user pose vector is \((3 \times 8) + 1 = 25\).

**Robot pose transformation**

The robot pose, denoted by \( r \), is represented by the joint angles of the upper body. Given that we utilized the Pepper robot in our experiments, we analytically computed 10 joint angles (Yu and Tapus, 2020): These angles include pitches of the hip and head, pitches and rolls of the left and right shoulders, and yaws and rolls of the left and right elbows (Robotics, 2020). As previously mentioned, the joint angles of the lower body were not taken into account due to concerns relating to balancing the robot’s body. Figure 4 showcases an example of a 3D skeleton model from the dataset, alongside its corresponding implementation in the Pepper robot.

**Neural network architecture**

Our proposed architecture consists of three main components: the encoder, decoder, and discriminator. Each of these components utilizes a long short-term memory (LSTM) unit, which is a form of recurrent neural network (RNN) that can effectively manage time-series data (Hochreiter and Schmidhuber, 1997). The encoder is fed a sequence of user poses \( U_t = u_{(t-m+1):t} \) as input, where \( m \) is set to 15, and the time interval between two adjacent user poses is 0.1 s. We configured the hidden state size of the LSTM in the encoder to 256. The outputs of the final LSTM unit are connected to a fully connected layer (FC), yielding a 128-dimensional vector representation \( z \) for the user poses. Furthermore, \( z \) was fully connected to another layer to generate the initial hidden states of the LSTM unit in the decoder.

The decoder accepts the current robot pose \( r_t \) as a seed pose and produces a sequence of robot poses \( R_t = r_{(t+1):(t+n)} \), where \( n \) was set to 5 during the training process. We set the hidden state size of the LSTM in the decoder to 512. The outputs of each LSTM unit were attached to a fully connected layer, which in turn emitted a set of 10 joint angle values to represent a robot pose. We incorporated skip (residual) connections from the input to all LSTM units in the decoder. As a result, the output of each fully connected layer represents the difference between the current and next poses (i.e., the change in the joint angle values). This setup ensures that the mean of the output values is close to zero, which in turn promotes faster convergence of the model during training. Finally, the decoder also generates the future robot behavior \( R_{t+l} = r_{(t+l+1):(t+l+n)} \), which serves as the input for the discriminator.

![Figure 4. Example of a 3D skeleton model in the dataset (left panel) and its implementation in the Pepper robot (right panel). (a) 3D skeleton model, (b) Pepper robot.](image-url)
The discriminator is designed to accept a sequence of robot poses as input and output the probability that the input pose sequence corresponds to a behavior from the training dataset. It is assumed that the robot behaviors from the training dataset are the ones that are appropriate (desirable) to be generated. Therefore, this framework encourages the discriminator to accurately distinguish between desirable and undesirable behaviors, which in turn assists the model in generating more appropriate behaviors. However, providing the next robot behavior \( \mathbf{R}_{t} \) as input to the discriminator might not yield effective results, because \( \mathbf{R}_{t} \) might not significantly differ from its initial pose \( \mathbf{R}_{0} \). Therefore, it becomes challenging for the discriminator to determine if the behavior is desirable. To overcome this issue, we feed the discriminator the future robot behavior \( \mathbf{R}_{t+l} \) where the model generation errors have accumulated over time, leading to a significant difference between the behaviors that the discriminator can discern. We empirically set \( l = n + 25 = 30 \) for the training process, which allows the discriminator to find noticeable differences between the generated behaviors. We configured the hidden state size of the LSTM in the discriminator to 512. The outputs of the last LSTM unit are connected to a fully connected layer, which ultimately outputs a single probability value.

**Training**

In our proposed neural network architecture, the combination of the encoder and decoder can be seen as the generator \( G \) of the GAN, with the discriminator \( D \) acting as the discriminator \( D \) of the GAN. During training, \( D \) learns to differentiate between real or fake (generated) behaviors, and \( G \) learns to generate realistic fake behavior that \( D \) will classify as real. Both models improve concurrently through competitive two-player game scenario. We define the loss function of \( G \) in such a way that it makes the robot’s generated behavior similar to the ground-truth behavior (first term), while ensuring \( D \) classifies it as real data (second term):

\[
\mathcal{L}_G = a_1 \cdot \text{MSE}(\langle \mathbf{R}_{t} \rangle_{gt}, \langle \mathbf{R}_{t} \rangle_{gen}) + a_2 \cdot \text{BCE}(D(\langle \mathbf{R}_{t+i} \rangle_{gen}), 1.0),
\]

where \( \text{MSE}(x, y) \) and \( \text{BCE}(x, y) \) are functions used to calculate the mean square error and binary cross entropy between two vectors \( x \) and \( y \), respectively. \( \langle \mathbf{R}_{t} \rangle_{gt} \) and \( \langle \mathbf{R}_{t} \rangle_{gen} \) are the ground-truth and generated values of \( \mathbf{R}_{t} \), respectively, and \( a_1 \) and \( a_2 \) are weighting parameters set to 100 and 10, respectively.

Furthermore, we define the loss function of \( D \) such that \( D \) classifies the ground-truth behavior as real data (first term) and the behavior generated by \( G \) as fake data (second term):

\[
\mathcal{L}_D = \beta_1 \cdot \text{BCE}(D(\langle \mathbf{R}_{t+i} \rangle_{gt}), 1.0) + \beta_2 \cdot \text{BCE}(D(\langle \mathbf{R}_{t+i} \rangle_{gen}), 0.0),
\]

where both \( \beta_1 \) and \( \beta_2 \) are set to 0.5. The \( \text{MSE} \) term, which brings \( \langle \mathbf{R}_{t} \rangle_{gen} \) closer to \( \langle \mathbf{R}_{t} \rangle_{gt} \) is included only in \( G \), and not \( D \). This is because \( G \) aims to make the generated behavior closely resemble the ground-truth, while \( D \) only aims to distinguish between real and fake data.

We trained \( G \) and \( D \) iteratively using the Adam optimizer (Kingma and Ba, 2014) with a mini-batch size of 100. The learning rate was set to 0.00001, and the gradient norm was clipped to a value of 1.0 to ensure stable training. The teacher-forcing technique (Bengio et al., 2015) was also implemented for more effective and efficient training, wherein the target robot pose was fed as the subsequent input to the LSTM unit of the decoder. For instance, we used the ground-truth output \( \langle r_{i} \rangle_{gt} \) of the \( h \)-th LSTM unit of the decoder as the input to the \((s + 1)\)-th LSTM unit instead of the generated output \( \langle r_{i} \rangle_{gen} \). The probability of using \( \langle r_{i} \rangle_{gt} \) was set to 0.5. This procedure allows the model to learn the input–output relationships of the training dataset more quickly through direct guidance.

**Experiments**

This section presents the results of the experiments performed using Pepper, a humanoid robot, in a simulated environment. The datasets used to train and test the neural networks are described in the *Training and test datasets* section. To validate the proposed behavior generation method, we describe examples of the behaviors generated in seven interaction scenarios in the *Validation of behavior generation* section. The results of a quantitative evaluation of the proposed neural network architecture are presented in the *Quantitative evaluation of neural network architecture* section.

**Training and test datasets**

To train the neural network architecture, we used the *AIR-Act2Act* dataset (Ko et al., 2021), which contains 5000 human–human interaction samples from 10 scenarios. Considering the complexity and stability of a real robot behavior implementation, seven interaction scenarios were selected, as listed in Table 1. From the selected interaction scenarios, the social behaviors learned by the robots were bowing, staring, shaking hands, hugging, and blocking face. As summarized in Table 1, 250 samples were used to train or test each interaction scenario, and the mean and standard deviation of the sample lengths were 187.5 frames (6.2 s) and 61.6 frames (2.1 s), respectively. The numbers of training and test data points extracted using the procedure described in the *Training data extraction* section, are presented in Table 2. From 1575 interaction samples, 116,462 training data points were extracted, and 12,738 test data points were extracted from 175 interaction samples. Each dataset containing the data for a particular interaction scenario performed by a particular person exists only once in either the training or test datasets.

The neural networks were trained for 300 epochs using the training data. To confirm that each training was complete, we qualitatively checked whether the model
generated realistic and diverse behaviors that responded to user behaviors and verified whether the generator and discriminator found a stable equilibrium by checking the loss values. Meanwhile, the test data were used to validate the behavior generation of the proposed model, as described in the following section.

Validation of behavior generation

To verify the proposed model, the behavior generation process depicted in Figure 5 was performed for each of the 175 test interaction samples. In the test phase, we set the values of \(m\) and \(n\) to 15 and 1, respectively, to ensure that the model produced instantaneous reactive behavior. For the first time step of behavior generation, the model takes the stack of user poses \(h_{u1:15}\) and the current robot pose \(h_{r15}\) as the input, and it generates the next robot pose \(h_{r16}^{gen}\). \(h_{r15}\) is used instead of a default robot pose as the model input for a fair comparison between the model generation results and the test data. In the next time step, the model generates the next robot pose \(h_{r17}^{gen}\) according to \(h_{r16}^{gen}\) and the stack of user poses \(h_{u2:16}\). We used \(h_{r16}^{gen}\) instead of \(h_{r16}\) because in a real robot operating environment, the current robot pose is the outcome of the behavior generation in the previous time step. Subsequently, the model repeats the generation of the next robot pose until the end of the test interaction sample.

Through this process, we tested the robot behavior generated in seven interaction scenarios. Figure 6 shows samples of the generated robot behaviors and key poses in each scenario. The poses in the top rows show the robot behavior \(B_{gt}\) extracted and converted from each test interaction sample, and the bottom rows show the robot behavior \(B_{gen}\) generated using the proposed method. The gray poses indicate the ground-truth robot poses extracted and converted from the test interaction samples. The teal pose represents the robot initial pose given as model input to start generating robot behavior, that is, \(h_{r15}\) in Figure 5. The blue poses represent the robot poses generated from the model, such as \(h_{r16}^{gen}\) and \(h_{r17}^{gen}\) in Figure 5. Note that the poses in Figure 6 were sampled such that the time interval between two adjacent poses was 0.5 s.

Previous studies demonstrated that key poses play an important role in behavior recognition (Chaaraoui et al., 2012; Dhiman and Vishwakarma, 2020); thick lines were used to indicate the key poses of \(B_{gt}\) and \(B_{gen}\), which

| Interaction scenarios | Human2 | #Samples | (#Frames) | (Seconds) |
|------------------------|--------|----------|-----------|-----------|
| Enters into the service area | Bows to Human1 | 250 | 296.6±51.3 | 9.9±1.7 |
| Walks around | Stares at Human1 | 250 | 171.1±26.1 | 5.7±0.9 |
| Stands still without a purpose | Stares at Human1 | 250 | 155.7±21.4 | 5.2±0.7 |
| Lifts arm to shake hands | Shakes hands with Human1 | 250 | 149.8±18.6 | 5.0±0.6 |
| Covers face and cries | Stretches hands to hug Human1 | 250 | 217.2±60.3 | 7.2±2.0 |
| Threatens to hit | Blocks face with arms | 250 | 149.9±20.5 | 5.0±0.7 |
| Turns back and walks to the door | Bows to Human1 | 250 | 172.1±35.2 | 5.7±1.2 |
| **Total** | | **1750** | **187.5±61.6** | **6.2±2.1** |

Table 2. Numbers of training and test data extracted from the selected interaction scenario data.

| Interaction samples | Training | Test | Total |
|---------------------|----------|------|-------|
| Extracted data      | 116,462  | 12,738 | 129,200 |

Figure 5. Process of generating robot behavior for each time step in the test phase. The gray box indicates the ground-truth user poses, while the teal and blue boxes indicate the ground-truth and generated robot poses, respectively.

Figure 6. Samples of the generated robot behaviors and key poses for each interaction scenario.
were qualitatively compared. In addition, each pose of the Pepper robot is displayed on the right with the key pose of \( \langle B \rangle_{\text{gen}} \) for each scenario. The pose with the maximum difference from the first pose was selected as the key pose for each behavior using

\[
k = \arg \max \sum \| p_j^i - p_j^0 \|, \quad j = 3, 6, 9
\]  

where \( p_j^i \) denotes the 3D position of the \( j \)th body joint of the \( i \)th robot pose with respect to the torso. The 3rd, 6th, and 9th body joints are the head, left wrist, and right wrist,

**Figure 6.** Samples of robot behaviors and key poses generated in seven interaction scenarios. (a) Scenario 1 (bowing). (b) Scenario 2 (staring). (c) Scenario 3 (staring). (d) Scenario 4 (shaking hands). (e) Scenario 5 (hugging). (f) Scenario 6 (blocking face). (g) Scenario 7 (bowing).
respectively, which play more important roles in social interactions than the other body joints. The position of each body joint is identified by solving the kinematic equation, where the lengths between the joints are set as follows, referring to Aldebaran Robotics (2020):

\[
\begin{align*}
d_1^2 &= 0.3 \text{ m}, \\
d_2^2 &= d_3^2 = d_4^2 = d_5^2 = d_6^2 = 0.15 \text{ m}, \\
d_7^2 &= d_8^2 = 0.08 \text{ m},
\end{align*}
\]

where \( k \) is the time index of the selected key pose and \( d_i^j \) is the distance between the \( i \)th and \( j \)th body joints.

The experimental results showed that the behaviors generated in scenarios 1, 4, and 5 had key poses on the same indices as the ground-truth behaviors, and the key poses were also similar. The key poses of the behaviors generated in scenarios 2 and 3 appeared at different indices; however, the entire pose sequence was almost identical to the ground-truth behaviors. In the behaviors generated in scenarios 6 and 7, the key poses appeared 0.5–1 s late or early, but because the key poses were similar, humans may consider them almost identical to the ground-truth behaviors.

In addition, we tested the robot behavior generated when a user lifted his/her right arm in four different positions to shake his/her hands. Figure 7 presents samples of the generated behaviors. We found that the robot shakes its hands by moving them up, down, left, and right according to the position of the user’s hand. In other words, the robot can respond to the user’s behavior by considering the posture.

**Quantitative evaluation of neural network architecture**

In this experiment, we compared the performances of different neural network architectures in the generation of robot social behaviors. Unlike manipulation or navigation problems, the success or failure of a social behavior task is ambiguous. The exact following of the ground-truth pose sequence is not the only solution. As shown in Figure 6(f), the key poses of \( B_{gt} \) and \( B_{gen} \) may not appear on the same index. However, because the key poses are similar, most users find that the two behaviors are nearly identical. Therefore, when evaluating the similarity between \( B_{gen} \) and \( B_{gt} \), it is necessary to consider the similarity between the key poses of the two behaviors as well as the similarity between the entire set of pose sequences. In particular, the movement of both hands and the degree to which the head is bowed are important for distinguishing social behaviors. It is also important for the robot to return to its initial position in a timely manner when generating long-term behavior.

Therefore, we used the existing RMSE-based metric \( S_1 \) and our defined metrics \( S_2 \) and \( S_3 \) to determine the difference between \( B_{gt} \) and \( B_{gen} \) as follows:

![Figure 7](image-url)
In other words, the metric architecture is denoted as the removing one of the architectural components used, and our architectures. The following four architectures were prepared by neural network architecture with those of different architectures started to overfit faster than the other architectures because their learning process was simpler.

First, the error in the entire pose sequence \( S_1 \) increased in the following order: Full network (0.009) \(<\) 3D position for user pose (0.009) \(<\) Without GAN loss (0.010) \(<\) Original GAN loss (0.011) \(<\) 3D vector for robot pose (0.012). We found that the proposed architecture outperformed other architectures by up to approximately 30%. The Original GAN loss and Without GAN loss architectures converged faster than the other architectures because their learning process was simpler.

The error in the key pose \( S_2 \) increased in the following order: 3D vector for robot pose (0.190) \(<\) 3D position for user pose (0.195) \(<\) Full network (0.199) \(<\) Without GAN loss (0.208) \(<\) Original GAN loss (0.229). Although there were no significant differences between the performances of the architectures, the proposed architecture outperformed the Original GAN loss and Without GAN loss architectures.

\[
S_1 = \text{RMSE}\left(\langle B \rangle_{\text{gt}} - \langle B \rangle_{\text{gen}}\right), \tag{8}
\]
\[
S_2 = \sum_{j \in \{3, 6, 9\}} \left\| \langle p^j \rangle_{\text{gt}} - \langle p^j \rangle_{\text{gen}} \right\|, \tag{9}
\]
\[
S_3 = \sum_{j \in \{3, 6, 9\}} \left\| \langle f^j \rangle_{\text{gt}} - \langle f^j \rangle_{\text{gen}} \right\|. \tag{10}
\]

where RMSE \((A, B)\) is the root mean square error between \(A\) and \(B\). \(\langle p^j \rangle_{\text{gt}}\) and \(\langle p^j \rangle_{\text{gen}}\) are the 3D positions of the \(j\)th body joint in the key poses of \(\langle B \rangle_{\text{gt}}\) and \(\langle B \rangle_{\text{gen}}\), respectively, and \(\langle f^j \rangle_{\text{gt}}\) and \(\langle f^j \rangle_{\text{gen}}\) are the 3D positions of the \(j\)th body joint in the final poses of \(\langle B \rangle_{\text{gt}}\) and \(\langle B \rangle_{\text{gen}}\), respectively. Metric \(S_1\) represents the sum of the errors between the poses generated at each time step. Metric \(S_2\) represents the sum of the distances between the head, left wrist, and right wrist in the two key poses, and metric \(S_3\) represents the sum of the distances between the head, left wrist, and right wrist in the two final poses. In other words, the metric \(S_1\) is for evaluating how similar the entire pose sequence is, the metric \(S_2\) is for evaluating how accurately the robot generated the key pose, and the metric \(S_3\) is for evaluating how timely and accurately the robot behavior is completed. To determine whether the robot behavior is well generated, all three proposed metrics should have small values because they evaluate different aspects of the robot behavior.

We used these metrics to compare the performance of our neural network architecture with those of different architectures. The following four architectures were prepared by removing one of the architectural components used, and our architecture is denoted as the Full network (proposed).

- **Original GAN loss** uses the next robot behavior \(\overrightarrow{R_i}\) instead of the future robot behavior \(\overrightarrow{R_{i+n}}\) as input to the discriminator. In other words, in Equations (4) and (5), \(I\) is set to 0 instead of \(n + 25\).
- **Without GAN loss** omits the second term in Equations (4) and (5) and trains only the generator \(G\), excluding the discriminator \(D\).
- **3D position for user pose** represents the user pose as the 3D positions of nine body joints (as in Equation (2)) instead of the direction vectors in Equation (3).
- **3D vector for robot pose** represents the robot pose as the direction vectors in Equation (3) instead of the joint angles.

Figure 8 shows the performance of the architectures evaluated using all test data (Table 2). Each architecture was trained using the same training scheme up to 300 epochs, and the errors in the entire pose sequence \(S_1\), key pose \(S_2\), and final pose \(S_3\) were observed every 10 epochs. The error curve is then drawn by calculating the three-observation moving average, representing the average of the three most recent observations. For a fair comparison between architectures, we compared the minimum error values of each curve, that is, the performance at which each architecture started to overfit on the training dataset.

Figure 8. Comparison between the performance of our neural network architecture and other architectures. (a) Moving average error in entire pose sequence \(S_1\), (b) Moving average error in key pose \(S_2\), (c) Moving average error in final pose \(S_3\).
In this study, we presented an end-to-end learning-based method for learning nonverbal behaviors from human–human interactions. We proposed a neural network architecture consisting of an encoder, decoder, and discriminator. In this framework, the encoder encodes the current user behavior, the decoder generates the next robot behavior according to the current user and robot behaviors, and the discriminator aids the decoder in producing a valid pose sequence after the long-term behavior is generated. The neural networks were trained using the human–human interaction dataset AIR-Act2Act. For this purpose, the user poses were extracted and normalized using the proposed vector normalization method, and the ground-truth robot poses were extracted and transformed into joint angles of the upper body.

To validate the proposed robot behavior generation method, experiments were performed using the humanoid robot, Pepper, in a simulated environment. The experimental results showed that the robot could generate five distinct social behaviors (i.e., bow, stand, handshake, hug, and block face) and adjust its behavior according to the posture of the user. Because it is difficult to assess success or failure in social behavior generation, we propose two metrics to compute the similarity between the generated behavior and the ground-truth behavior. Using these metrics, we showed that the network architectural components used (i.e., the GAN-based loss functions, use of future behavior as input for the discriminator, user pose normalization, and robot pose transformation) improved the performance of robot behavior generation. Moreover, the proposed method was able to learn seven social behaviors without significantly degrading the performance, which is a significant result that has not yet been studied.

When a robot generates these nonverbal social behaviors, users feel that their behavior is understood and emotionally cared for. Consequently, these nonverbal social behaviors can be applied not only to home service robots, but also to social robots in spaces where users feel emotionally cared for.

### Table 3. Comparison between performance of learning a single behavior and that of learning all seven behaviors.

| Error in | Scenario 1 - bowing | Scenario 2 - starring | Scenario 3 - staring | Scenario 4 - handshaking | Scenario 5 - hugging | Scenario 6 - blocking face | Scenario 7 - bowing |
|----------|----------------------|-----------------------|----------------------|-------------------------|-----------------------|------------------------|----------------------|
| S1       | 0.011                | 0.002                 | 0.000                | 0.008                   | 0.012                 | 0.011                  | 0.008                |
| Head     | 0.078                | 0.011                 | 0.000                | 0.022                   | 0.023                 | 0.036                  | 0.088                |
| LWrist   | 0.116                | 0.016                 | 0.001                | 0.035                   | 0.049                 | 0.089                  | 0.095                |
| RWrist   | 0.081                | 0.021                 | 0.001                | 0.059                   | 0.069                 | 0.077                  | 0.074                |
| S2       | 0.275                | 0.048                 | 0.002                | 0.116                   | 0.141                 | 0.202                  | 0.257                |
| Final pose | 0.092              | 0.013                 | 0.001                | 0.020                   | 0.022                 | 0.019                  | 0.060                |
| Head     | 0.041                | 0.016                 | 0.001                | 0.036                   | 0.041                 | 0.041                  | 0.042                |
| LWrist   | 0.046                | 0.016                 | 0.001                | 0.045                   | 0.027                 | 0.030                  | 0.036                |
| RWrist   | 0.179                | 0.037                 | 0.002                | 0.101                   | 0.090                 | 0.090                  | 0.138                |

The representation methods for the user and robot poses did not affect the key pose errors of the generated behaviors. However, their advantage is that they consume only a small amount of computational resources because the number of features is small.

Finally, the error in the final pose ($S_3$) increased in the following order: Full network (0.091) ≈ 3D position for user pose (0.094) < 3D vector for robot pose (0.104) < Original GAN loss (0.117) ≈ Without GAN loss (0.119). The proposed architecture outperformed the other architectures by approximately 30% because it used GAN-based loss functions to generate competent long-term behavior.

In the next experiment, we compared the performances of learning multiple behaviors versus a single behavior as shown in Table 3. We used the full neural network architecture, but trained it on data from only one interaction scenario each time. For each interaction scenario, the model with the smallest $S_1 + S_2 + S_3$ value was selected from among the models trained for 300 epochs. The errors in the entire pose sequence ($S_1$) and final pose ($S_3$) are similar when learning a single behavior and all seven behaviors. The error in the key pose ($S_2$) when learning all seven behaviors increased by approximately 30% compared with when learning a single behavior. However, considering that the network learned the data seven times, this is not a large value. Moreover, the errors in the positions of the head, left hand, and right hand were 5.9 cm, 7.0 cm, and 7.1 cm, respectively, which are reasonable for social interactions (Prasad et al., 2021).
but also to guide robots, delivery robots, educational robots, and virtual robots, enabling users to enjoy and effectively interact with them. However, as this study aimed to generate the behavior of a humanoid-type robot, additional research is required to retarget the behavior so that it can be applied to other types of robots. Additionally, as the lower body of the robot was not considered in this study because of the balancing problem, further studies are needed to generate behaviors that include lower body movements, such as moving forward or backward. We also intend to conduct further experiments to test a robot’s ability to exhibit appropriate social behaviors when deployed in the practical world and facing a human; the proposed behavior generator would be tested for its robustness to noisy input data that a robot is likely to acquire. Moreover, by collecting and learning more interaction data, we plan to extend the number of social behaviors and complex actions that robots can exhibit. Furthermore, it is expected that the performance can be further improved through attempts such as using the transformer-based model proposed in Vaswani et al. (2017). Finally, we would like to conduct further research on a one-to-many generative model so that a single human behavior can be mapped to multiple robot responses by referring to Sun et al. (2020), Chen et al. (2020), Yu and Tapus (2020), and other studies. If the robot can learn various behavioral policies considering the user’s age or cultural differences, more natural human–robot interaction can be obtained.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was partly supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No.2017-0-00162, Development of Human-care Robot Technology for Aging Society, 50%) and (No.2020-0-00842, Development of Cloud Robot Intelligence for Continual Adaptation to User Reactions in Real Service Environments, 50%).

ORCID iDs

Woo-Ri Ko https://orcid.org/0000-0001-7534-8835
Minsu Jang https://orcid.org/0000-0002-7166-0300
Jaeyeon Lee https://orcid.org/0000-0002-9494-1359
Jaehong Kim https://orcid.org/0000-0002-6840-5026

References

Ahn H, Ha T, Choi Y, et al. (2018) Text2action: generative adversarial synthesis from language to action. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). Brisbane, QLD, USA: IEEE, pp. 5915–5920.

Arakawa R, Kobayashi S, Unno Y, et al. (2018) DQN-tamer: human-in-the-loop reinforcement learning with intractable feedback. arXiv preprint arXiv:1810.11748.

Bengio S, Vinyals O, Jaitly N, et al. (2015) Scheduled sampling for sequence prediction with recurrent neural networks. arXiv preprint arXiv:1506.03099.

Breazeal C and Scassellati B (1999) How to build robots that make friends and influence people. In: Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No. 99CH36289). Kyongju, Korea: IEEE, pp. 858–863. volume 2.

Buckcshash H and Raman B (2020) Variational conditioning of deep recurrent networks for modeling complex motion dynamics. IEEE Access 8: 67822–67834.

Chaaraoui AA, Climent-Pérez P and Flores-Revuelta F (2012) An efficient approach for multi-view human action recognition based on bag-of-key-poses. In: International Workshop on Human Behavior Understanding. Berlin, Germany: Springer, pp. 29–40.

Chen W, Wang H, Yuan Y, et al. (2020) Dynamic future net: diversified human motion generation. In: Proceedings of the 28th ACM International Conference on Multimedia. New York, NY, US: Association for Computing Machinery, pp. 2131–2139.

Cho K, Van Merriënboer B, Gulcehre C, et al. (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Dhiman C and Vishwakarma DK (2020) View-invariant deep architecture for human action recognition using two-stream motion and shape temporal dynamics. IEEE Transactions on Image Processing 29: 3835–3844.

Dindo H and Schillaci G (2010) An adaptive probabilistic approach to goal-level imitation learning. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. Taipei, Taiwan: IEEE, pp. 4452–4457.

Goodfellow I, Pouget-Abadie J, Mirza M, et al. (2014) Generative adversarial nets. Advances in Neural Information Processing Systems 27: 2672–2680.

Hochreiter S and Schmidhuber J (1997) Long short-term memory. Neural Computation 9(8): 1735–1780.

Hua M, Shi F, Nan Y, et al. (2019) Towards more realistic human-robot conversation: a seq2seq-based body gesture interaction system. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Macau, China: IEEE, pp. 1393–1400.

Huang CM and Mutlu B (2012) Robot behavior toolkit: generating effective social behaviors for robots. In: 2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI). Boston, MA, USA: IEEE, pp. 25–32.

Jonell P, Kucherenko T, Ekstedt E, et al. (2019) Learning non-verbal behavior for a social robot from youtube videos. In: ICDL-EpiRob Workshop on Naturalistic Non-verbal and Affective Human-Robot Interactions, Oslo, Norway: Engineer’s House Conference Centre

Karpathy A, Toderici G, Shetty S, et al. (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. New York, NY, USA: IEEE, pp. 1725–1732.
Kingma DP and Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Ko WR, Lee J, Jang M, et al. (2020) End-to-end learning of social behaviors for humanoid robots. In: 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC). Toronto, ON, Canada: IEEE, pp. 1200–1205.

Ko WR, Jang M, Lee J, et al. (2021) AIR-Act2Act: human–human interaction dataset for teaching non-verbal social behaviors to robots. The International Journal of Robotics Research 40(4-5): 691–697.

Levine S, Pastor P, Krizhevsky A, et al. (2018) Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research 37(4-5): 421–436.

Makatchev M, Simmons R and Sakr M (2012) A cross-cultural corpus of annotated verbal and nonverbal behaviors in receptionist encounters. arXiv preprint arXiv:1203.2299.

Mitsunaga N, Smith C, Kanda T, et al. (2008) Adapting robot behavior for human–robot interaction. IEEE Transactions on Robotics 24(4): 911–916.

Pham H, Liang PP, Manzini T, et al. (2019) Found in translation: learning robust joint representations by cyclic translations between modalities. Proceedings of the AAAI Conference on Artificial Intelligence 33: 6892–6899.

Prasad V, Stock-Homburg R and Peters J (2021) Learning human-like hand reaching for human-robot handshaking. arXiv preprint arXiv:2103.00616.

Raessa M, Chen JCY, Wan W, et al. (2020) Human-in-the-loop robotic manipulation planning for collaborative assembly. IEEE Transactions on Automation Science and Engineering 17(4): 1800–1813.

Rahmatizadeh R, Abolghasemi P, Bölöni L, et al. (2018) Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). Brisbane, QLD, Australia: IEEE, pp. 3758–3765.

Rapantzikos K, Avrithis Y and Kollia S (2009) Dense saliency-based spatiotemporal feature points for action recognition. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. Miami, FL, USA: IEEE, pp. 1454–1461.

Redmon J and Farhadi A (2017) Yolo9000: better, faster, stronger. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. New York, NY, USA: IEEE, pp. 7263–7271.

Robotics A (2020) Pepper - documentation. https://doc.aldebaran.com/2-8/home_pepper.html. [Online].

Salem M, Kopp S, Wachsmuth I, et al. (2012) Generation and evaluation of communicative robot gesture. International Journal of Social Robotics 4(2): 201–217.

Salem M, Eyssel F, Rohlfing K, et al. (2013) To err is human (-like): effects of robot gesture on perceived anthropomorphism and likability. International Journal of Social Robotics 5(3): 313–323.

Shotton J, Fitzgibbon A, Cook M, et al. (2011) Real-time human pose recognition in parts from single depth images. In: CVPR 2011. Colorado Springs, CO, USA: IEEE, pp. 1297–1304.

Singh T and Vishwakarma DK (2019) Video benchmarks of human action datasets: a review. Artificial Intelligence Review 52: 1107–1154.

Singi S, He Z, Pan A, et al. (2023) Decision making for human-in-the-loop robotic agents via uncertainty-aware reinforcement learning. arXiv preprint arXiv:2303.06710.

Srivastava S, Li C, Lingelbach M, et al. (2022) Behavior: benchmark for everyday household activities in virtual, interactive, and ecological environments. In: Conference on Robot Learning. Atlanta, GA, USA: PMLR, pp. 477–490.

Sun G, Wong Y, Cheng Z, et al. (2020) Deepdance: music-to-dance motion choreography with adversarial learning. IEEE Transactions on Multimedia 23: 497–509.

Sutskever I, Vinyals O and Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems. Cambridge, MA, USA: The MIT Press, pp. 3104–3112.

Tuyen NTV, Georgescu AL, Di Giulo I, et al. (2023) A multimodal dataset for robot learning to imitate social human–human interaction. Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction. Stockholm, Sweden: IEEE, 238–242.

Vaswani A, Shazeer N, Parmar N, et al. (2017) Attention is all you need. Advances in Neural Information Processing Systems 30: 5998–6008.

Wada K and Shibata T (2007) Living with seal robots—its sociopsychological and physiological influences on the elderly at a care house. IEEE transactions on robotics 23(5): 972–980.

Wang P, Li W, Gao Z, et al. (2015) Action recognition from depth maps using deep convolutional neural networks. IEEE Transactions on Human-Machine Systems 46(4): 498–509.

Weitschat R and Aschemann H (2018) Safe and efficient human–robot collaboration part ii: optimal generalized human-in-the-loop real-time motion generation. IEEE Robotics and Automation Letters 3(4): 3781–3788.

Wu X, Xiao L, Sun Y, et al. (2022) A survey of human-in-the-loop for machine learning. Future Generation Computer Systems 135: 364–381.

Yang PC, Sasaki K, Suzuki K, et al. (2016) Repeatable folding task onto, ON, Canada: IEEE, pp. 1200–1205.

Zaraki A, Wood L, Robins B, et al. (2018) Development of a semi-autonomous robotic system to assist children with autism in developing visual perspective taking skills. In: 2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN). Nanjing, China: IEEE, pp. 969–976.