Virtual Reality (2022) 26:1519–1538
https://doi.org/10.1007/s10055-022-00644-4

ORIGINAL ARTICLE

Immersive machine learning for social attitude detection in virtual reality narrative games

Georgiana Cristina Dobre[1] · **Marco Gillies**[1] · **Xueni Pan**[1]

Received: 19 March 2021 / Accepted: 12 February 2022 / Published online: 7 April 2022

Abstract
People can understand how human interaction unfolds and can pinpoint social attitudes such as showing interest or social engagement with a conversational partner. However, summarising this with a set of rules is difficult, as our judgement is sometimes subtle and subconscious. Hence, it is challenging to program Non-Player Characters (NPCs) to react towards social signals appropriately, which is important for immersive narrative games in Virtual Reality (VR). We collaborated with two game studios to develop an immersive machine learning (ML) pipeline for detecting social engagement. We collected data from participants-NPC interaction in VR, which was then annotated in the same immersive environment. Game design is a creative process and it is vital to respect designer's creative vision and judgement. We therefore view annotation as a key part of the creative process. We trained a reinforcement learning algorithm (PPO) with imitation learning rewards using raw data (e.g. head position) and socially meaningful derived data (e.g. proxemics); we compared different ML configurations including pre-training and a temporal memory (LSTM). The pre-training and LSTM configuration using derived data performed the best (84% F1-score, 83% accuracy). The models using raw data did not generalise. Overall, this work introduces an immersive ML pipeline for detecting social engagement and demonstrates how creatives could use ML and VR to expand their ability to design more engaging experiences. Given the pipeline's results for social engagement detection, we generalise it for detecting human-defined social attitudes.

Keywords Artificial intelligence · Expressive body language · Gaming · Human–computer interaction · Virtual agents · Virtual reality

1 Introduction

Complex human behaviours exhibited in everyday social interaction are hard to recognise automatically and therefore to use as mechanics in videogames.

As a result, players often find themselves driving a social interaction in a videogame by choosing what to do from a menu (see Sect. 3.1 for examples). In Virtual Reality (VR), this could break the plausibility illusion (Slater 2009) and lead to break-in-presence (Slater and Steed 2000), which takes the players back to the real world and significantly reduces the level of immersion. In this work, we explore a novel pipeline in game design, combining Machine Learning (ML) and VR, with the aim to make social interactions in VR narrative games more engaging, immersive and inclusive (in the sense that it will appeal to a broader audience than current video games).

VR devices could enable richer input mechanisms than that of a traditional videogame. In non-VR games, often players are limited to 2D user interfaces (keyboards, 2D game-controllers). In VR, users can deploy a diverse range of motions in 3D: they can use their limbs, head, or their whole body as a form of input to drive the interaction, as they would do in their day-to-day life.

One of the most promising uses of body movement in VR is social interaction with Virtual Characters (VCs), or Non-Player Characters (NPCs). In face-to-face interactions with people, we use our bodies extensively as non-verbal communication (colloquially called 'Body Language'), including actions such as gaze (eye contact), gestures, posture and the use of personal space. VR opens the possibility to use these social cues as first-class elements of gameplay and thus creating much richer social experiences in games.

✉ Georgiana Cristina Dobre
c.dobre@gold.ac.uk

[1] Department of Computing, Goldsmiths, University of London, London, UK

However, when the user input is more complex than button-pressing, it is a challenge to interpret its meaning in real time. Rule-based methods work well for detecting certain social behaviours when the hands and head need to be in a certain position and/or rotation (e.g. raising hands or looking at something). On the other hand, more complex social behaviours are more difficult to detect using fixed rules, and we might even judge the same situation differently due to our personality and expectations. Examples of these social behaviours are social attitudes, which in this paper refers to a feeling towards another person expressed through verbal and particularly non-verbal behaviours taking place in social interactions, such as sympathy, affection, aggression, or social engagement. These social attitudes are too complex to be identified using a set of rules; however, people can identify them in human–human (or human–VC) interactions.

Nevertheless, there are clear benefits to replace traditional *explicit interactions* (selecting an option by clicking a button) with *implicit interactions* (social attitudes expressed via body language), where the player's non-deliberate, *implicit* actions are inputs for a game (Schmidt 2000). This is in particular important for maintaining the plausibility illusion in character-driven narrative games in VR where players could engage with NPCs naturally. Furthermore, being able to explore the use of social attitude with NPCs as a possible game mechanic (as opposed to, for instance, shooting NPCs with a gun) also makes games more inclusive and appealing to a wider demographic than current videogames.

In this work, we collaborated with two immersive game studios to create a pipeline for detecting the social attitude engagement that could be used as implicit interaction in a narrative VR game. The detected social engagement can then be used to trigger different behaviours in the NPC or in the game environment itself, influencing how the game continues. These triggers are to be decided by the game designers and game creators, based on how they envision the game and the gameplay. For instance, if a player is detected to be socially engaged with an NPC, they could gain higher trust score from this NPC and could display animations that reflect a higher level of social engagement in return. The developed pipeline is meant to be integrated in the game studio's animation process in order to use the predictions to animate the NPC or the game environment. The pipeline is independent of any company-dedicated software as the set-up (Fig. 1 part A, B) can be recreated and the algorithms used are available for implementation in other software.

In this work, we focused on the social engagement detection part, the NPC's response to the user's social engagement in the game being out of our scope. We also argue that the pipeline used here to detect social engagement can be generalised and used for other social attitudes such as sympathy, affection or aggression.

Game design is a creative process that involves the design of mechanics that guide players into certain desired behaviour patterns. While it is important that these behaviours in some way reflect players' natural inclinations, they are also defined by game designers who may want to guide players away from their more common patterns of behaviour. This is particularly true of the scenario we are studying in this paper, as there was an explicit desire to guide players away from traditionally anti-social behaviour in narrative games towards pro-social interaction. We therefore view the work on social engagement detection as a creative interaction design process. Game designers should be in control of how the game, and characters, in particular, respond to different actions in a player, just as, in traditional games, designers are in control of how the game responds to button presses. The definition of social engagement should not be viewed as

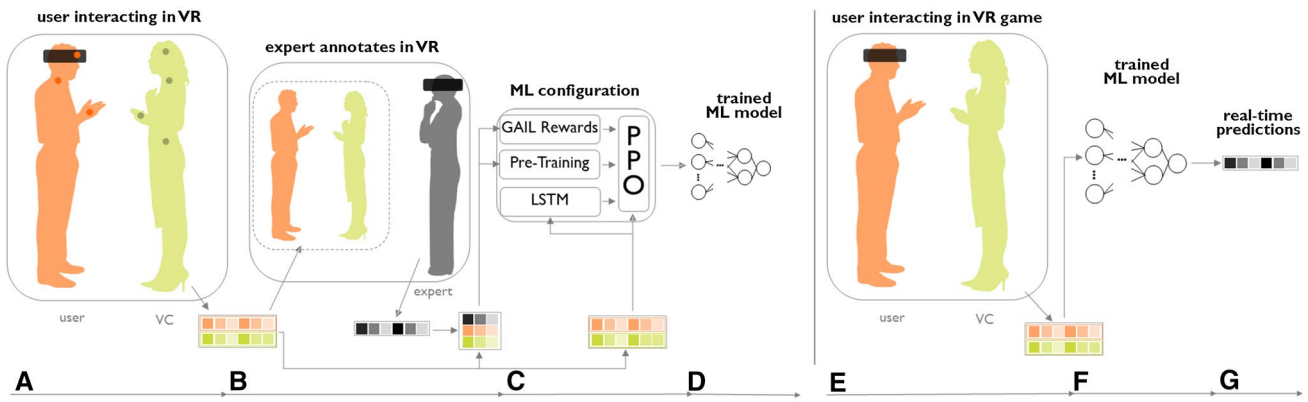

Fig. 1 Pipeline for detecting human-defined social engagement, including immersive data collection (user interaction (**A**) and expert annotating (**B**)) for training the machine learning model. This takes place by pre-training the model, creating Generative Adversarial Imitation Learning (GAIL) rewards for the reinforcement learning algorithm Proximal Policy Optimisation (PPO) that also uses a temporal memory called Long Short-Term Memory (LSTM) algorithm (**C**). This process exports a trained ML model (**D**). In a user–VC interaction (**E**), the trained model (**F**) detects in real time the human-defined social attitude (**G**) which could be used in different scenarios

an attempt to capture some objective measure (as might be done in traditional machine learning) but as a reflection of the game designer's creative judgement. The integration of machine learning into the creative process of game design and the foregrounding of creative judgement is one of the main contributions of this work.

The above-mentioned factors that social attitudes are largely subconscious, that the behaviour is implicit and that this forms part of a creative project (a game), create a situation that we believe is relatively little studied. We are attempting to recognise a concept with no clear explicit definition. Social engagement and certainly the behaviours associated with it are highly variable and contextual. If we were to attempt a definition it would be far higher level than the detail needed for computational implementation. There is also no ground truth. Biometric measures might be used in some emotional context, but can only really distinguish low-level physiological states such as arousal, not high-level cognitive/emotional/social concepts like engagement. So we are dealing with a concept that can be defined only implicitly through human judgement. It might be possible to use player's own judgement of their feelings while interacting with a character, but these may well not correspond well to their outward behaviour, it is perfectly possible for a person to be interested in what another is saying without outwardly displaying it, or conversely to outwardly appear highly engagement while inwardly feeling bored and thinking of other things (a fairly common human behaviour pattern). More importantly, the use of player's own annotations would compromise the creative process. As described above how players interact with the game should be the result of a design process led by creative judgement. In this work we therefore treat the definition of social attitudes as a creative process driven by the judgement of a game designer. Social attitudes are, in this paper, therefore concepts without explicit definition or ground truth and defined solely through expert creative judgement. This type of interaction design concept will be increasingly common as VR becomes a medium used by creative practitioners and which attempts to tap more complex and subtle aspects of human behaviour. Machine learning is particularly well suited to this task as it does not require an explicit definition at any point, simply a set of examples, which can be created through creative judgement. This is the key aim of this paper.

In the following, we underline our aim and contributions (Sect. 2). Then we review some of the related work around non-verbal behaviours for VCs/NPCs and attitudes detection with a focus on social engagement (Sect. 3). Next we describe how the pipeline is used for detecting social engagement in Sect. 4, where we also cover the experimental study with the data collection and the annotation process in VR. In Sect. 5 we describe how the model is trained, covering different input data and model configurations; then we present our results in Sect. 6. Section 7 explains how the pipeline could be generalised to be used as a detection tool for other social attitudes. We cover the limitations and discussion in Sect. 8 and conclude in Sect. 9.

2 Aim and contributions

As we are collaborating with two games companies, we aim to develop a workflow which supports the creative design process and can be implemented into a production-ready VR game for the consumer market. After several workshops with our creative partners, we identified our three key challenges at the beginning of the process: (1) *Gamer Behaviour*: this is part of a product which will be available on the market. Thus *it has to work for most gamers* (who will be paying for the game), which is very different from experimental studies with paid participants in the laboratory we were more accustomed to. (2) *Creative Process*: not only do we want to automatically detect a complex social attitude in real time, but the judgement also has to be part of the creative process. In the game industry, Creative Directors are the 'superstars' who define the artistic design of a game—we need to include them as much as possible in this process. (3) *Market Reach*: The game has to be accessible to as many players as possible, meaning it will be developed cross-platform, considering the most commercially available headsets. This also means we are limited to the consumer market VR Headsets inputs (i.e. no access to eye, mouth, or EEG trackers) and software platforms that are compatible with major games consoles.

In order to tackle the first challenge, we need to better understand the *gamer behaviour*. After several in-depth discussion we learnt from our industry partners that although players usually talk to other players in an online game, they almost never directly talk to NPCs. Thus, we needed to create a scenario where a specific social attitude (social engagement) could be present without the user speaking to the NPC. Through multiple brainstorming sessions (see supplementary material Figure S2, we identified *social engagement* as a suitable social attitude to detect, as it could be present merely as a *listener* behaviour. It is also suitable for their current game in production, where the player has to gain trust from various NPCs as part of their mission.

To address the challenge of detecting complex social behaviours and making it part of the *creative process* (in which the creative director could be involved as much as possible) we decided to explore the method of *imitation learning* where the ML algorithms could learn from their creative director. We taught an ML model to imitate how a human would judge the social interaction (Fig. 1). This is because as humans we can easily detect the level of social engagement without always being able to verbally describe

it, and different people might make different judgements for the same setup of behaviours due to their individual experiences. The ML model was trained on human annotations of the interaction between the user and the NPC in VR. The annotation also took place in VR, making it an immersive process (Fig. 3). This enhances the annotator's capability of observing the interaction from multiple angles and moving around the scene freely in the recorded interaction.

To make the game *accessible to a broad market*, we were limited to develop the pipeline with data captured from the player's headset and hand-controller. As we collaborated with two games companies, we were restricted by their game engine platform. We designed the data collection study and trained the ML model within the Unity3D game engine (www.unity.com), using Unity ML-Agents. Further, we needed to use less complex ML models to reduce the computational cost, increasing the successful deployment and usage on all VR consumer devices (such as Oculus Quest, PSVR and PC-powered VR devices). This would allow the game to work real time while on these platforms, maintaining frame rate required for running VR.

We chose an imitation learning approach, rather than, for example, supervised classification in order for the method to fit more broadly within the framework of virtual agent behaviour used in industry. While supervised learning focuses on learning direct mappings between an input and output, reinforcement and imitation learning methods learn policies: probability distributions on the actions agents take in particular circumstances. A policy determines which actions should be taken in a given state of the world and the agent and therefore is a direct driver of the agent's behaviour. This focus on actions taken rather than mappings makes it well suited to modelling the behaviour of agents. Reinforcement learning is, for this reason, the most commonly used learning approach in the games industry. This makes it appealing in our context for two reasons (Shao et al. 2019). Firstly, it is the most familiar approach in the games industry and is therefore more likely to be adopted. Secondly, it is more readily extensible to more complex agent behaviour models, which might not be the case for supervised learning. However, reinforcement learning per se is not suitable for this application, since it requires a well-defined measure of success or failure to use as a reward signal. In a standard game the score or win/lose condition can be used; however, this does not apply to social interaction. Instead we use imitation learning in which the reward signal is determined based on how well the agent's behaviour matches a human demonstration.

The main contribution of this work is the introduction of a creative director-focused pipeline for machine learning of social engagement detection that can be used for other social attitudes (see Sect. 7). This pipeline provides three principle novel contributions:

First, we designed and conducted an experimental study of an immersive data collection process in which participants listened to an NPC's monologue (prepared by professional writers) in a VR environment closely resembling a real game social interaction.

In three different VR stages, we gave participants either no instructions (*VR stage 1*), instructions that would very likely lead to socially engaged (*VR stage 2*), or socially disengaged behaviours (*VR stage 3*). Results from this experiment not only gave us useful insights on how players could behave in a VR game but also provided data to train our ML algorithms. Participants without instructions did not normally engage in social interactions, showing the benefit of providing realistic game tasks to guide behaviour during data capture (see Sect. 4.3).

Secondly, we developed an immersive environment where game designers could annotate the captured data, identifying instances of the social engagement. This VR environment placed the annotator in the same virtual space as the participant and the VC, enabling them to watch the interaction as if it were a real-life conversation. This allows them to make the most effective use of their social cognition and also creates an artist-friendly environment for data annotation, which is close to real gameplay experiences. This latter turns data annotation from a technical task to one that benefits from an interaction design skill.

Finally, with our pipeline, we were able to train an ML model to detect implicit social engagement in VR interactions with 83% accuracy. Specifically, we used a reinforcement learning algorithm with imitation learning rewards from examples set by human experts.

We report our comparison between the model's configurations and features that worked well. Our results show that pre-training the ML model improved the performance as did the use of temporal memory via an LSTM network. In addition, we propose several psychologically derived data features as inputs to the training which we show generalise better than raw features.

3 Related work

3.1 Virtual characters in narrative games

Narrative or story-driven games are those with a clear storyline where the players' actions are based on the story and can influence it (Ip 2011). In these type of games, the game mechanics are not only centred around actions performed in the game, but also in the story behind the gameplay. The narrative function in a game creates compelling and engaging play as it borrows aspects from other forms of narrative media (such as film and literature), adding emotional depth to the player's experience.

In many narrative games, the VCs (or NPCs) are a core element. Often players can interact with them (fight/get help from) or even have a dialogue with them as part of the gameplay. The outcome of the interaction often leads directly on to the next actions available, making the interaction itself part of the game mechanics. Hence, the narrative genre games are designed around the overall story and the player's interaction with the NPCs. Examples of games that fall into this category include: *The Walking Dead Series* (https://www.skybound.com/telltales-the-walking-dead-the-definitive-series), *Heavy Rain* (www.quanticdream.com/en/heavy-rain), *Mass Effect* (www.ea.com/en-gb/games/mass-effect) or *L.A. Noire* (www.rockstargames.com/lanoire). In most cases, these games are non-VR and dialogues with the NPCs involve players selecting phrases from a pre-defined list, using the buttons from mouse, keyboard, or joysticks.

Recent years have seen the rise of VR games which push the game engagement to the next level. Some of these games applied the classic game mechanics and interaction methods directly from non-VR games onto the VR ones (*Hellblade: Senua's Sacrifice VR* www.hellblade.com/), others attempted to adapt some of the interactions to 3D controllers. For instance, in *Moss* (www.playstation.com/en-gb/games/moss-ps4/), players could use the PlayStation controllers to navigate and interact and are directly involved in the narrative by controlling the main character (a young mouse called *Quill*) from a third-person perspective. Although these games are more immersive than non-VR games, using VR does not necessarily lead to improved user experience (e.g. simulation sickness in Christensen et al. 2018).

We argue that it is key to enable natural interaction utilising the richer inputs VR offers. Because users can move freely in VR, the interaction in these games does not have to be restricted by the game controllers. The user's large and diverse range of inputs can be manipulated to design interactions with VCs that are closer to the ones taking place in real life. This aspect helps maintain the user's plausibility illusion, which means that the user's experience of interacting with a VC is similar to an interaction that happens face to face with a real person.

There are popular VR games that make good use of natural interactions, such as *Beat Saber* (https://beatsaber.com/) or *SuperHot VR* (https://superhotgame.com/vr/). However, most of them are not centred on a story, nor the interaction with NPCs. *Dance Central* (http://www.dancecentral.com/) is another popular VR game where players dance based on instructions, mimicking dance movements from VC instructors. Although there are many NPCs whom the users can interact with, the interaction itself is done through a virtual mobile phone.

It is more difficult to develop narrative games in VR with natural social interactions. This is because the natural interactions with NPCs are more complex than the interactions in non-narrative settings (such as slicing cubes with lightsabers—in Beat Saber). Other VR games, such as *Half-Life Alyx* (https://www.half-life.com/en/alyx/), implement ways of interacting with the environment that are very close to how people do in daily life. Being able to open doors by pushing them, manually reloading weapons, crawl and freely move around, enhances users' feeling of presence. However, most of the games like this one, rely on core mechanics such as shooting or fighting, making them violent. Having these violent behaviours happen in VR can have a strong and profound effect on the players' emotion and behaviour (Wilson and McGill 2018; Bailenson 2018), thus excluding users less interested in violent or action-based games.

The games industry is trying to find other ways of designing interaction and other game mechanics that would better fit the VR medium. Our collaboration aims to aid the creation of first-person VR narrative games that make use of the VR technology and that is not centred around traditional game actions but rather on social interactions.

3.2 Modelling non-verbal behaviour for VC

Literature suggests that in most cases, VCs' non-verbal behaviour is generated through statistical modelling, rule-based or by making use of ML models. These approaches lead to autonomous VCs or semi-autonomous ones (partly controlled by a human).

In statistical modelling, the VC's behaviour is generated based on probabilities from human-to-human interactions such as gaze behaviour based on speaking or listening roles (Lee et al. 2002). Rule-based methods use simple hard-coded algorithms: the non-verbal behaviour is often pre-captured and played back based on a set of pre-defined rules (Marsella et al. 2013). This is sometimes incorporated with a Wizard-of-Oz setup, where an assistant steers the VC's non-verbal behaviour by using a predefined set of buttons (Pan and Hamilton 2018). Lastly, ML models learn non-verbal behaviours from a large amount of data and use them to drive different aspects of the VC's non-verbal behaviour in social interactions (Ferstl and McDonnell 2018; Greenwood et al. 2017).

These methods perform differently based on the type of interaction they are applied to. In a structured task scenario (where the user's actions are limited to specific ones), they tend to perform well. However, in free-flow scenarios, with no pre-defined structure or fixed actions, statistical modelling and rule-based models struggle, while ML methods tend to show better results (Forbes-Riley et al. 2012; Dermouche and Pelachaud 2019a; Jin et al. 2019).

The rapport between the user and VC is essential, especially when it comes to unstructured situations in social interactions. For instance, in a medical doctor training (Pan et al. 2018), the non-verbal behaviours link to the overall

rapport, while the doctors' non-verbal behaviours influence the patients' perception of their empathy (Brugel et al. 2015). In social interactions, people adapt and adjust their verbal and non-verbal behaviours based on their partner's behaviour and the overall social interaction (Burgoon et al. 2006). Works such as Dermouche and Pelachaud (2019b), Ahuja et al. (2019) or Feng et al. (2017) take into account data from all participants in that interaction to detect or generate different aspects of a social interaction; however, social aspects (attitudes) between the user and the VC also influence the interaction dynamics.

Training an ML model with the data from all participants in a social interaction better illustrates it, leading to a more robust outcome when compared to data from only one participant. This is because, during dyadic human-human social interactions, one person's behaviour highly influences the behaviour of the other person (Steed and Schroeder 2015; Burgoon et al. 2006). Moreover, humans behave differently depending on whom they are interacting with, their culture or upbringing and whether they are by themselves or in someone else's presence (Schilbach et al. 2013).

Taking into account this aspect in human–VC interactions makes the VC's behaviour flexible and able to adapt to the scenario at hand. Being able to detect different interaction dynamics and attitudes between the user and VC could be used to develop behavioural models. These drive the VC in social interactions, its non-verbal behaviour being dependant on the rapport/empathy between the user(s) and the VC (Cafaro et al. 2016).

3.3 Detecting social attitudes

Detecting attitudes in social interactions could be a complex task for a machine to undertake with using only a set of rules. Rules are not able to cover the intricacy of interaction dynamics. However, it comes naturally to us, humans, to recognise and interpret complex non-verbal behaviours, even from a still image (Vinciarelli et al. 2011).

There is an increasing body of literature on detecting different social attitudes such as dominance, agreement, or engagement in interactions (Dermouche and Pelachaud 2019a; Khaki et al. 2016; Bee et al. 2009). These tackle the interaction from video recordings and could be applicable to VCs on 2D displays. They make use of features such as prosodic information, gaze direction, turn-taking or facial expressions (action units). Though these studies are influential contributions to the field, they are not directly applicable in VR. This is because not all user's features are traceable (e.g. facial expressions) and because the interaction in VR has more dimensions available (e.g. proximity) that are missing on a 2D screen.

Social engagement is an important aspect to consider during user–VC interactions. As with other social attitudes, the VC should adapt its behaviour with a change in the engagement level. This has been researched on many occasions, for instance in Gordon et al. (2016), Woolf et al. (2009), Bohus and Horvitz (2014), Dhamija and Boult (2017). They propose methods that tackle engagement in interactions; however, they disregard the user–VC interaction dynamics loop. Dermouche and Pelachaud (2019a) include this loop in their work, detecting the engagement from dialogue videos on a 5-level engagement scale. They also assess the ML models that use only one person's data; however, these models show lower performance than the one considering both people's data. They trained the model on actions units (AUs), head rotation, gaze angle and the conversational state of the interaction. They report that the AUs feature has the highest contribution to the model's performance. When trained on this alone, the model's performance is 98%, improving to 99% when all features are used.

The work proposed here is similar to Dermouche and Pelachaud (2019a) as the user–VC interaction dynamics loop is taken into account, as well as considering a temporal ML model (LSTM) to detect social engagement. However, our medium is different (VR vs. 2D screen), and we examine different features. The features in their work cannot be reproduced here because the user's head is covered by the head-mounted display. Thus, the feature with the highest contributions (AUs) is not available in this setup. These studies define social engagement, usually in a different way from one another, as there are many engagement definitions (Glas and Pelachaud 2015). In this work, we do not use a specific definition of engagement, rather the annotator defines the social engagement behaviour through annotating the social interaction within the recording setup (in immersive VR).

To detect social engagement in social interactions, we propose a pipeline based on *imitation learning*. We introduce a method to integrate natural social interaction aspects as game mechanics in narrative VR games. The method is based on synchronised data from both interaction participants (NPC and the user), as it would happen in the final game. Through this, the game can detect social engagement and trigger an action that would make the game progress without the user's explicit input. We propose that the pipeline can be also generalised to social attitudes detection.

4 Method: social engagement detection

Here, social engagement broadly refers to the social engagement one shows in social interactions linking it to the action of paying attention and showing interest. However, social engagement is a complex and subjective social attitude that is difficult to be described using concrete rules. Humans, on the other hand, have the ability to easily identify when social engagement takes place. Since this understanding is

implicit, and we are designing a machine learning process based on creative judgements not on an objective definition, we do not formally define social engagement. Instead, the concept emerges implicitly from the annotator's judgement of participants' behaviour. In this section, we describe how to detect social engagement between a user and a VC in an immersive VR scenario using the ML pipeline from Fig. 1.

In the next part of this section, we detail how we used the pipeline to collect data for detecting social engagement. We collected the data from users and then from the annotator. These processes happened separately but both in VR.

We first describe the scenario we designed especially for this data collection process (Sect. 4.1), then data collection with participants (Sect. 4.2), followed by how the annotation was done (Sect. 4.3). Finally, in Sect. 4.4 there is an overview of the human's annotations and the questionnaire result (Sect. 4.5).

4.1 The scenario for data collection

We created an immersive and interactive VR scenario where users' behaviour can be recorded. Specifically, users can interact with a VC (See Fig. 2 and in supplementary material, Figure S1) created using Adobe Fuse Software (www.adobe.com/uk/products/fuse.html) and rigged & animated using Mixamo (www.mixamo.com). This interaction took place in a room that we designed to resemble a bedroom that will be used in the game, as suggested by the game company we were working with (see Figure S3 in supplementary material). The user can interact with, grab or change the location of the majority of objects in the room, for example vanity box, birdcage, pillow, flower and vase, books, bin or chair, but not others, such as picture frame, poster, candle, rug, room divider.

4.1.1 Virtual Character Implementation

During the interaction, the VC carried out a monologue about her family and her life. We collaborated with a national centre for immersive storytelling where professional writers wrote a captivating script. The monologue was pre-recorded and played back for each VR stage. Table 1 illustrates part of the monologue. While performing the monologue, the VC carried out animations for different behaviours as described in the monologue script from the professional writers. The VC performed generic scripted animations (such as: look at bird cage, point to the door) using inverse kinematics to express specific behaviours. See supplementary videos for the monologue animation.

4.1.2 Study Design

The study took place in VR and contained three stages based on the user instructions, which aimed to trigger both high and low social engagement behaviours in users. In the first VR stage ($S1$) the user was told to interact with the environment and the VC as they would do in a gameplay, allowing us to study the range of different behaviours that participants would perform without prompting, gaining insights into the type of gameplay behaviours we could expect. In the second VR stage ($S2$), the user received instructions to try to gain the VC's trust, representing the kind of task players would be given in the game. This VR stage aims to record mostly high social engagement data. For the third and final VR stage ($S3$), the user received instructions to

Fig. 2 Example of users interacting with the VC. On the left, the user is listening while looking directly at the VC. In the middle, the user is patting the VC on the shoulder. Based on the user's questionnaire after the session, the user is interacting with the VC by trying "[...] *to comfort her* [the VC] *by touching her shoulder when she was emotional* [...]". In the image on the right, the user is interacting with objects in the environment; in this case, the user is swinging a birdcage

Table 1 A snippet of VC's monologue. The text in italic represents the scriptwriter's indications. As an interactive monologue, the user was directly addressed to in sections such as *Do you think they would have found a new home?* For the monologue animation, see supplementary videos

Wistful monologue spoken with a sombre tone
VC: That's the only place we could laugh freely. The park with the rose finches. They've built apartments on it now. No longer can I ever go there. I wonder what happened to all the finches? Maybe they found a new home
VC stares directly at the player again, her brow slightly crumpled
VC: Do you think they would have found a new home?
VC shakes her head briefly and her shoulders slump over a little bit
VC: No, they're like me, still looking for somewhere else to call home. I often imagine them happy[...]

explore the room, representing a typical task that players would be familiar with from other games. The interaction from *S*3 aimed to produce primarily low social engagement data. All tasks were designed based on feedback from our game developer partners to represent typical gameplay. For an example of the participants' behaviour in each part, see video in supplementary material.

The three VR stages took place in the same order for all users: *S*1, *S*2 and then *S*3. Since we are not comparing different stages, counterbalancing is not required. It was also not possible to counterbalance since doing Stages 2&3 before Stage 1 could prime participants to be either social or anti-social and therefore affect their performance in Stage 1.

Ahead of *S*1, participants explored a training room that was similar to the room in the experiment, where they could interact with objects (open drawer/doors, grab objects) and move around the room. This extra step ensured the users were comfortable with the VR headset, navigation and VR interaction techniques. All three VR stages and the training step took place in VR with an Oculus Rift Headset. It took about 5 min for each VR stage, resulting in each participant spending about 20 min in VR, with a small break between each VR stage when they filled in questionnaires.

4.2 User data collection in VR

There were in total 13 participants, 9 males and 4 females, aged between 20 and 46 years and an average of 32 years old. In terms of VR experience, 31% used VR less than 10 times, 38% more than 10 times but less than 50 and 31% more than 50 times. All participants voluntarily agreed to take part in the experiment and signed a consent form. The whole process was approved by the University's ethics board.

The data collection took place in two batches because of time and participants availability restrictions. The first batch was with 6 participants and the second with 7. The only difference between the first and second batch is in the VC's location and gaze direction (see Figure S4 in supplementary documents). This difference was introduced to investigate the effect of the agents' gaze at various key objects; however, this did not give significant results and will not be discussed in this paper. Nonetheless, this did allow for a more diverse dataset, where the VC had more than one location and variable head and body orientations.

The term *session* refers to each time the participant took part in the virtual scenario (regardless of the VR stage), hence, there are three sessions for each participant. There is a missing session from S3 in the second batch due to a software error, resulting in a total of 38 sessions, with 18 sessions from the first batch (6×3) and 20 from the second ($7 \times 3 - 1$). In total, the time spend in the VR environment by all participants is approximate 190 min (38 sessions $\times$ 5 min per session).

The experiment run on Unity3D and we collected data from both users and the VC. As described in Table 2,

Table 2 Data were recorded from participants and VC; the head and hands are relative to the corresponding root of each VC and the user; 3D vectors represent the X, Y and Z components in a vector data structure; the Quaternion represents the X, Y, Z, W rotation components

Information recorded	Data type
User's head position	3D Vector
User's head rotation	Quaternion
User's left- and right-hand position	3D Vector
User's left- and right-hand rotation	Quaternion
User's main head root position	3D Vector
User's main head root rotation	Quaternion
User's left & right index and hand triggers	Float
User's headset velocity & angular velocity	3D Vector
VC's head position	3D Vector
VC's head rotation	Quaternion
VC's left- and right-hand position	3D Vector
VC's left- and right-hand rotation	Quaternion
VC's main root (hip) & chest position	3D Vector
VC's main root (hip) & chest rotation	Quaternion

we recorded head, hands and root positions and rotations from the VC and the user. The root for the VC was situated in the hip and in the head for the user. The root is not same for the user and VC because the character model used was structured differently. Apart from that, we also collected the user's index and trigger buttons from the controller. They were using these buttons to grab objects in the scene. And lastly, we collected the user's headset velocity and angular velocity to capture the user's motion. We chose to collect the position and rotation data to record where the user and the VC are in the scene and where they are facing. The user's and VC's non-root (hands and head) information is relative to the root data as these elements are "children" of the root element in the Unity hierarchy. These data are then mapped in between −1 and 1 to meet the Unity ML-Agents recommended best practice (see Sect. 5.3). Because we used these values straight from the trackers as they were available in Unity3D engine, we refer to this dataset as *raw data*. Although there is no clear definition of raw data in the literature, in this paper we use this notation to refer to the unaltered version of the data.

In total, over 108, 000 frames of multi-modal data were used to train and evaluate the ML model (see Sect. 5 and Table 4).

4.3 Human annotations in VR

A human annotator watched a playback of the user interacting with the VC and annotated their interaction.

As *social engagement* is a very subjective term and it has many definitions (Glas and Pelachaud 2015), a human annotator marked the data without directly defining social engagement. In this case, the annotator implicitly defined social engagement by annotating it during the user–VC interactions.

The annotator labelled the sessions' playbacks in random order. They did not know which VR stage or which user they were annotating.

To assure the annotator had rich social interaction information, they could access the user's and the VC's camera view (showing their current viewpoint). This allowed the human annotator to have access to exactly what they were viewing at any time while being in the same place as the user and the VC. An example of this is seen in Fig. 3A. Different hand controller buttons ('A' and 'B') switched on/off the user's or the VC's camera view. The annotator marked the beginning of the high or low social engagement period, using the other hand controller buttons ('Y' and 'X', respectively, Fig. 3B). As they pressed 'Y' or 'X', the '−' or '+' signs coloured for 0.5 s with the correspondent colour (red or green). The '−' or '+' signs were on the annotator's (virtual) hand side.

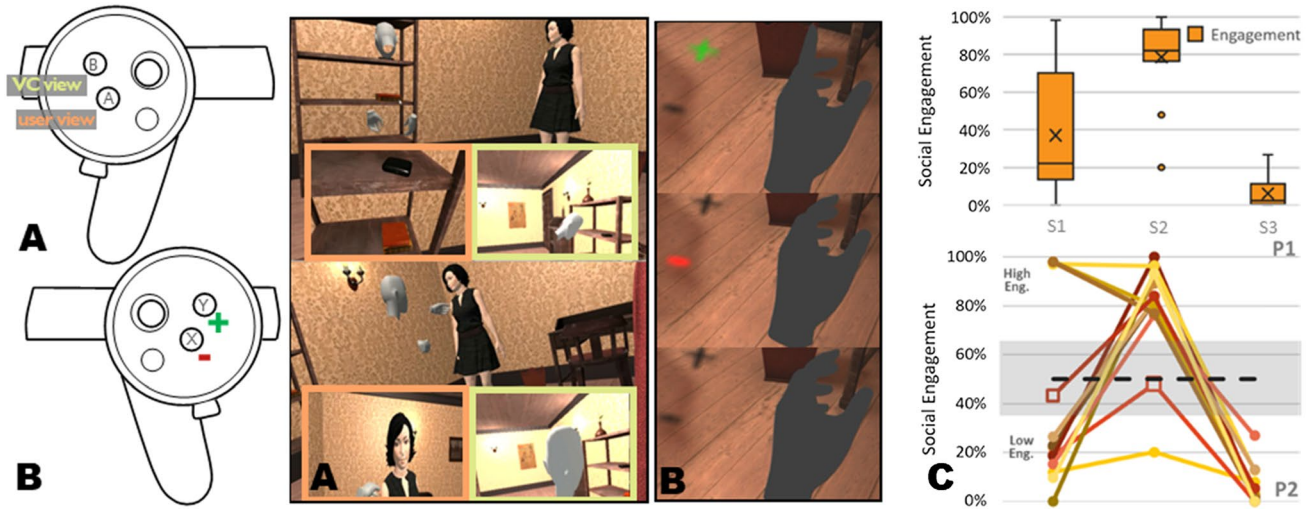

Fig. 3 Expert's annotations: **A** Controls mapping the camera view: the 'A' and 'B' buttons act as a switch to activate/deactivate the camera view from the user's or VC's perspective. **B** Controls mapping the social engagement level: 'X' and 'Y' record the current social engagement level rating, illustrated by colouring for 0.5 s the red '−' or the green '+' signs next to the (virtual) hand. **C** Engagement marked by the human annotator: **P1** shows the average (x) and median (line) percentage of all 38 sessions by the VR stage. These VR stages took place in the same chronological order (S1, S2, S3) for all participants. **P2** illustrates the average percentage of all sessions by the VR stage and by participant. The back dotted line shows the 50% threshold that delimits the high from low social engagement sessions. Each other coloured line represents one participant. Sessions with a large mixture of low/high social engagement are positioned on a grey background and marked with a square

The participant's avatar was represented in a simple way which showed only the head and hands in an abstract form (see Figs. 2 and 3A). This was important as it removes features that were not accessible by the ML algorithm. Instead, the playback displayed only representations reconstructed from the data collected from the players. Therefore, it ensured that the annotator was not making judgements based on features that were inaccessible to the ML algorithm and thus cannot be learned by it. Gillies et al. (2015) give examples of this problem. Annotators used video to annotate motion, but the learning algorithm used motion capture data. The result was that annotators (consciously or subconsciously) detected different behaviours based on features such as muscle tone or facial expression that were not available to the algorithm, which was therefore not able to learn to distinguish the movements. Although facial and eye information are relevant when it comes to social attitude detection (see Sect. 3), in this work we focused particularly on body gestures. This is because of the technical limitations imposed by the HMDs available in the current VR consumer market. We also decided not to include voice because each player could have very different background noise and different accents (making recognition challenging and unreliable), and we were informed by our game industry collaborators that gamers do not normally talk to NPCs (see the *Gamer Behaviour* and the *Market Reach* challenges in Sect. 1). Furthermore, the most important features from the literature (Sect. 3) such as gaze and body posture are strongly related to the feature we chose: head and body movements.

4.4 Annotations overview

We computed the percentage of high/low social engagement labels between the user and the VC from each session by each VR stage. We calculated it as a percentage of all high (respectively, low) social engagement frames over the total number of frames.

As expected, when the users received instructions to behave with high or low social engagement (*S*2, respectively, *S*3), the users acted accordingly. For VR stage *S*2, the mean of high social engagement is 79%. Similarly, for VR stage *S*3 the mean percentage of high social engagement is 6%. For the VR stage *S*1, however, the behaviour is mixed: most users showed low social engagement with some users displaying high social engagement. Figure 3C-P1 shows these averaged levels over the three VR stages for all 38 sessions. In Fig. 3C-P2, these values are separated by each user, showing their behaviour in each session. The grey background colour highlights the sessions with a mix of high and low social engagement. Most of the sessions have the expected engagement level (most of *S*2 recording high social engagement and *S*3 low social engagement level) with two exceptions for *S*2. Many users showed low social engagement when not given an instruction (*S*1). Although most of the averaged session's social engagement can be categorised as *low* or *high*, there are two sessions (from *S*1 and *S*2) that are very close to the 50% threshold (marked with a black dotted line). These two sessions have a square marker in the Fig. 3C-P2.

4.5 Questionnaire results

Participants answered a few questions after each VR stage. These questions were customised to the VR stage they just experienced. They could also leave some free comments about that stage.

4.5.1 VR Stage 1

After *S*1 (where they would hear the monologue for the first time without any instructions), they were asked to answer questions about the VC, such as: to list the family members the VC was talking about, the relationship the VC has with her family and how they think the VC was feeling; they were also allowed to write any comments about this stage.

Three participants wrote that they did not listen to the VC and another six that they stopped listening after a while; these participants had an incomplete or wrong list of family members, or wrote that the VC's relationship with her family is '*loving, good memories*' (the VC was talking about the affair her mother had with her uncle and how her father did not come to her mother's funeral). Based on the annotator's marking, these participants had either a low social engagement score, less than 15% (the ones who said they did not listen) or between 19–26% and one score of 43% for those mentioning they stopped listening after a while. The remaining four participants were able to answer the question about the family members correctly, or almost correctly (two of them missed the mother, and one even mentioned the finch—the bird that the VC was talking about). In the general comments part, those participants also wrote about the way they perceived the monologue and what they think of the VC. Based on the human annotator, these participants had over 97% social engagement score.

4.5.2 VR Stage 2

Here, we instructed the participants to gain the VC trust. After this VR stage they were asked how well they performed at gaining the trust and also to write any comments regarding this VR stage. The majority of them (11 out of 13) described what they tried to do to gain the VC's trust. They said they listened, tried to be empathetic, nod when appropriate and '*stopped messing around*' because '*If I start looking through drawers and cupboards, I think she would be more suspicious of me*'. These participants got

over 76% high social engagement score based on the annotator's marking. One of the remaining two participants said they were *'expecting some "helpers" to point out what you can or can't do to a character'* and that they could not earn her trust by themselves. This participant's score of social engagement was 48%. The last participant wrote that they did not interact with the VC at all which reflects their low score of 20% given by the human's annotations.

4.5.3 VR Stage 3

After the last VR stage, where the instruction was to explore the room and remember as many objects as possible, the participants were asked to list all items they recall and to write any comments they have about this VR stage. All of them described how they explored the room and how that felt like: for some, it felt more immersive than the previous sessions, for others it was the opposite: *'having full control on exploring I lost a bit of immersion as I was behaving as I wouldn't do in the real world'*. Others mentioned that the VC did not comment on them exploring the room (the VC having the same monologue as in the first sessions) or that they found '*it more interesting to interact with the objects while she speaks about them, (looking at the birdcage when she talks about the finches)*'. All participants reported a high number of items (from 9 to 17, with an average of 14), while in the room there were 23 items. As expected, the annotator gave low social engagement scores to all participants in this VR stage, as it can be seen in Fig. 3C, P1 and P2.

In summary, the results for $S1$ show that participants had a range of different behaviours when they were not prompted with a particular task, but with the majority biased towards low engagement. $S2$ and $S3$ were successful in generating the desired behaviour, using realistic gameplay tasks. This shows the benefit of giving participants tasks to implicitly guide their behaviour (though the inclusion of unprompted behaviour could still be useful to identify unexpected behaviour patterns).

5 Training the detection component

We trained the model using imitation learning with the Unity ML-Agents platform (v0.11) and their main reinforcement learning algorithm Proximal Policy Optimization (PPO). In this section we explain the algorithms used (Sect. 5.1), then in Sect. 5.2 we cover the ML configuration, followed by what input data we considered (Sect. 5.3) and ending with Sect. 5.4, the ML implementation.

5.1 ML algorithms

We proposed different model structures, including pre-training with recorded data and adding temporal memory through a recurrent neural network (Long Short-Term Memory: LSTM). Below we present a brief description of each model.

PPO (Schulman et al. 2017) is a Reinforcement Learning algorithm and the idea behind these algorithms originated from behavioural psychology. It refers to an agent that changes its behaviour to maximise a reward function. The goal of a reinforcement learning algorithm is to develop a policy. This policy maps states to probabilities of selecting a certain action, with the aim of maximising the expected reward. More specifically, PPO trains a stochastic control policy where the agent learns its behaviour from experience without prior information on the environment or the task. Here, *stochastic* refers to having a probability distribution associated with all actions from each state.

To ensure that the model has a good starting point for optimisation, we pre-train the model using behaviour cloning (a simpler imitation learning model than the Generative Adversarial Imitation Learning, described below). This uses the training examples to find a good initial set of weights for the neural network for the full training algorithm.

Generative Adversarial Imitation Learning (GAIL) (Ho and Ermon 2016) is an approach to imitation learning, learning to execute a task by imitating human performance. It does this by generating reward signals from a human performance that are used to train the PPO reinforcement learning algorithm. It relies on data usually provided via human (or expert) demonstrations to learn a policy that behaves similarly to the human. The algorithm compares state-action pairs (each input and current circumstance with the corresponding response) from expert data against state-action pairs generated using the policy. In the same time, a classifier trains to differentiate expert data from the generated one. Thus, the policy develops to generate data that the classifier would mistake for the expert data.

Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) is a recurrent neural network. It learns a series of events with time order that have long time intervals. Based on this, it automatically determines the optimal time lags (time between 2 consecutive events), used for the next prediction. Its neural network is composed of one input layer, one output layer and one recurrent hidden layer. The recurrent layer contains a memory block structure that memorises the temporal state and controls the information flow. It learns how past actions unfolded, thus knowing when to incorporate or drop past events and take future decisions.

5.2 Proposed ML configurations

PPO provides positive rewards for performing the desired behaviour and negative ones for the non-desired behaviours. In this case, to mimic an imitation learning scenario, the rewards are calculated using GAIL. These rewards show the performance on the action the model took and influence future actions.

We hypothesise that both pre-training it and adding a temporal memory (through LSTM) would improve the PPO's performance. The behaviour learned from pre-training influences the action taken by PPO, at the same time, PPO's policy attempting to maximise the reward. The temporal memory takes into account past actions; hence, the algorithm considers past behaviour and current actions when deciding what to do next (what action to take). We hypothesise this because the behaviour that needs to be learnt is complex and temporal. We compare these models with those without (using random initialisation instead of pre-training and a standard feedforward network instead of LSTM).

5.3 Input data for proposed ML training

There is strong evidence in the literature that certain behaviour aspects (such as body posture or gaze) are linked to social engagement (Mota and Picard 2003; Sanghvi et al. 2011). Based on this, we trained the model with psychologically based features, such as the user's facing direction, distance from the VC, interaction with other objects and their velocity and angular velocity (as shown in Table 3). Since these are calculated from the raw data we collected, we call them derived data.

The user's distance from the VC is calculated based on Hall's personal space (Hall 1966). We calibrate the virtual space units using the average human's height of 1.65 m (Max Roser and Ritchie 2013) and mapped it to the user's height in the virtual space units from the VR headset. Hall's personal space has three different space layers: intimate (0.4 m), personal (1.1 m) and social space (3.6 m). From these three, we use the intimate and social spaces as the lower and higher boundary. We calculate these thresholds from the VR headset height, which we assume it represents the average person's height (1.65 m). Therefore, the 0.4 m intimate space threshold is calculated from height/4 = 0.4 m; and the 3.6 m social space threshold from height/0.45 = 3.6 m. The values are then mapped between 0 and 1. Thus, 0 is the most further away from the VC: the maximum and above of *social space* and 1 is the closest to the VC: minimum of *intimate space* and below.

The use of derived data features inspired by the psychology of social interaction has the potential to improve ML performance. However, recent trends in Deep Learning have shown that deep neural networks are able to learn effective representations directly from raw data (Bengio et al. 2013). In our evaluation, we, therefore, compare models trained on derived data with those trained directly on raw data. The raw data used here was pre-processed to comply with the Unity ML-Agents best practice. We mapped each element of the $3D$ vector raw values from the maximum and minimum positions possible to values between -1 and 1. This comparison was done by training the best performing model configuration on both raw data and derived data, detailed in Table 2.

Both models (derived and raw data) use the human's annotations as ground truth data, and its output is a discrete binary value. The discrete value shows the current user's social engagement at each frame. It can have the value of 1, for the user's high social engagement, or -1, for the user's low social engagement.

These two options aim to mimic the human's ratings of low/high social engagement during the annotation. To label the data, the human presses buttons for high or low social engagement behaviour; the data in between the button-presses represent the most recent pressed value. For instance, if the annotator marks *high* at time t_i, *low* and time t_{i+1}, and then *high* again at t_{i+2}, all timeframes from t_i to t_{i+1} are labelled high social engagement, all timeframes from $t_i + 1$ to t_{i+2} are low social engagement and everything from t_{i+2} until the next button pressed is high again. This way, the annotations are in the same format and frequency as the model's output, generating a value for each frame. The data have a frequency of 9 to 10 frames per second. We decided on this frequency as it has been used in the literature for low-level and subtle behaviour such as fast head nods (Hale et al. 2020).

Table 3 Derived input data. These are calculated based on the raw data detailed in Sect. 4.2 Table 2

Description	Data type
Distance between the user and VC, based on Hall's personal space (Hall 1966), value mapped between 0 and 1	Float
User's facing direction: the angle between VC's head rotation and user's head rotations divided by 180	Float
Interaction with objects: data from the controllers' trigger (the trigger allows objects interaction)	Float
User's headset velocity	3D vector
User's headset angular velocity	3D vector

The dataset for training with both raw and derived input data is detailed in Table 4, also showing the percent of the annotated low and high social engagement for all three VR stages. There are fewer dataframes in the last VR stage ($S3$) as there is a missing session due to software error. The missing session does not unbalance the dataset because the expected social engagement from that VR stage is low engagement, and there are already more than half low social engagement sessions from $S1$ (see Table 4 and Fig. 3C-P2).

5.4 Implementation

We analysed two additions to the PPO ML structure: pre-training and a temporal memory via LSTM. Therefore, we compare the PPO algorithm implemented with different configurations: with and without pre-training it, and with and without LSTM.

We randomised the dataset sessions and divided it into three folds of 13, 13 and 12 sessions each, for a 3-fold cross-validation; the training data consists of two folds while the remaining one represents the evaluation data. The hyper-parameters are tuned for both models with derived and raw input data (see Table $S1$ in supplementary material). The hyper-parameters corresponding to *LSTM* and *pre-training* are dropped for the training configurations where these models are not used (in PPO+GAIL+LSTM, PPO+GAIL+PreTrain or PPO+GAIL).

6 Results

In this section, we present the results of the presented pipeline. Section 6.1 covers how the model's prediction data is post-processed to match the format of the ground truth data. In Sect. 6.2 we present the results of the models trained with derived features, while in Sect. 6.3 we provide the comparison of the model trained with derived features and the model that uses the raw dataset.

Table 4 Proportions of **High** and **Low** social engagement annotated data used for all three VR stages for training the model. In this table, *Ann.* is short for *Annotations* and $S1-3$ for each VR stage

	Sessions	Data frames	High ann. %	Low ann. %
S1	13	37,226	37.1	62.9
S2	13	37,288	78.5	21.5
S3	12	33,931	6.1	93.9
Total	38	108,445	41.5	58.5

6.1 Data post-processing

We post-process the data in two different occasions: (1) we smoothed out the model's predictions data to remove noise and (2) we averaged the model's predictions and the ground truth data from a 1-s section. The latter process returns one value for each section, which will be used to compare the model's predictions to the ground truth data. We detail the post-processing actions in the remaining of this section.

6.1.1 First data post-processing

We describe how the human annotates the ground truth data in Sect. 4.3. Briefly, the annotator marks only the change in the social engagement (from low to high or from high to low engagement); thus, the ground truth data contain large blocks of either low or high social engagement data. The ML model outputs the predictions in a different way: it predicts a social engagement value at each frame. In many cases, this can result in noisy output, with regions of low social engagement containing a few frames of high social engagement (or vice versa). For instance, if we take a segment of length 10 dataframes (approx. 1 s), it can contain a majority of high engagement values, say 8, the remaining 2 being low engagement values. If we would compare frame-by-frame, the 2 low social engagement values are in minority in that window, and they can be seen as noise. When evaluated against the ground truth data, the 2 dataframes would appear as false negatives if the whole window would have high social engagement values.

Because of this difference between how the annotator created the ground truth data and how the models output the predictions, we post-process only the model's output data to remove the noise.

We smooth it out by applying a rolling window (with evenly weighted points) of 0.5 s on the model's outcome (see Eq. 1). This results in a float value; because it is not compatible with the ground truth data (integer datatype), we average the result to 1 if the rolling window result is higher than 0 and to -1 otherwise (Eq. 4). The post-processing is further explained below:

Generically, a rolling window can be represented as:

$$W_i^{j,h} = \{x_{i-j}, \ldots, x_i, \ldots, x_{i+h}\} \quad j, h \epsilon \mathbb{N}. \tag{1}$$

The number of samples in $W_i^{j,h}$ being: $|W_i^{j,h}| = j + h + 1$. The notations j and h refer to the number of items to consider for the window that appear on the left side, respectively, on the right side of i. For a 0.5s window size on a 10fps frequency, the number of samples is 5, hence the values for j and h could be 2 and 2, respectively, creating a symmetric window centred in x_i.

Given X containing all dataframes from a session, such as:

$$X = \{x_1, x_2, \ldots, x_n\} \tag{2}$$

a window can be represented as:

$$W_i^{2,2} = \{x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}\}, \quad i\epsilon[3, |X| - 2] \tag{3}$$

Then, the value for a dataframe (x_i) from X is:

$$x_i = \frac{1}{\left|W_i^{2,2}\right|} \sum_{x_m \epsilon W_i^{2,2}} x_m \begin{cases} -1, & x_i \leq 0 \\ 1, & x_i > 0 \end{cases} \tag{4}$$

6.1.2 Second data post-processing

Although the ML model outputs a value at each time frame, the social engagement is very unlikely to switch from one state to another and then back to the initial state in a very short period of time (one-tenth of a second). Similarly, in other studies, the authors consider certain time sections. For instance, Yu et al. (2004) manually divide the conversation in utterances and use those for prediction and in Bohus and Horvitz (2014), they consider a 5-s section for forecasting disengagement.

We take a similar approach and average both ground-truth data and the predicted data over a time of 1 s. With this, we compare the model's output to the ground-truth data and calculate the performance metrics.

We calculate the mean value from each time section, then we round the result to use: 1 if the mean is greater than 0.49; 0 if the mean is in between −0.49 and 0.49; and −1 if the mean is smaller than −0.49. Thus, given the time section S_t:

$$S_t = (x_t, x_{t+1}], \quad t\epsilon\mathbb{N}, \quad t\epsilon[0, T] \tag{5}$$

and T is the session length in seconds, then the value V of each section is:

$$V_{section} = \frac{1}{|S_t|} \sum_{x_m \epsilon S_t} x_m \begin{cases} 1, & V_{section} > 0.49 \\ 0, & -0.49 \leq V_{Section} \leq 0.49 \\ -1, & V_{section} < -0.49 \end{cases} \tag{6}$$

The results have three categories: 1 for *High* social engagement, 0 for *Mix* social engagement and −1 for *Low* social engagement. The *Mix* social engagement appears when a time section contains very similar numbers of *High* (1) and *Low* (−1) datapoints, such that the average on that time section is greater than −0.49 but lower than 0.49 (as in equation above). In the ground truth data, this tends to happen at transitions between low and high, but in the prediction data it can also happen when the model is not very stable, the output fluctuating from one social engagement rating to another. These are the three categories for all model's confusion matrices as seen in Tables 5 and 6.

To compute the performance, we compare each rounded window value from the true data to the corresponding time window in the predicted dataset. We evaluate all trained models based on accuracy and F1-score metrics. Accuracy is a measure that shows how often the model's output is correct. F1-score (Chinchor 1992) measures how well a model performs, combining precision and recall by their harmonic mean (Eq. 7). Precision is the number of true positives (true data that is predicted as being true) divided by the number of true positives plus the number of false positives (true data that is predicted as being false), while recall is the number of true positives divided by the number of true positives plus the number of false negatives (true data that is predicted as being false):

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{7}$$

6.2 Model configurations

We consider different model configuration for training the model with the derived data (Table 3). We compare these configurations to test our assumption that the temporal model (LSTM) and/or pre-training improve the performance of detecting social engagement. We performed these tests using derived input data related to social engagement, as described in Sect. 5. The results for each configuration are calculated using a 3-fold cross-validation method (two folds for training, one fold for testing).

A repeated two-way ANOVA conducted in SPSS (version 24) indicated both LSTM and pre-training had a significant effect on the accuracy and F1-score (accuracy - LSTM: $F_{(1,37)} = 58.52, p < 0.001, \eta^2 = 0.613$, pre-training: $F_{(1,37)} = 386.12, p < 0.001, \eta^2 = 0.913$; F1-score - LSTM: $F_{(1,37)} = 14.83, p < 0.001, \eta^2 = 0.286$, pre-training $F_{(1,37)} = 412.74, p < 0.001, \eta^2 = 0.918$), there is also an interaction effect LSTM x pre-training (accuracy $F_{(1,37)} = 11.13, p = 0.002, \eta^2 = 0.231$, F1-score $F_{(1,37)} = 7.57, p = 0.009, \eta^2 = 0.170$). This means both LSTM and pre-training have significantly improved the result, and both should be used at the same time to get the best results.

Figure 4A shows the variance of accuracy and F1-score metrics on different model configurations trained on derived data; the numbers on the figure represents the averages of these. As hypothesised, the configuration with both LSTM and pre-training performs the best in terms of accuracy and F1-score average (83.4% and 84.1%). The second best is the configuration where the model is pre-trained but

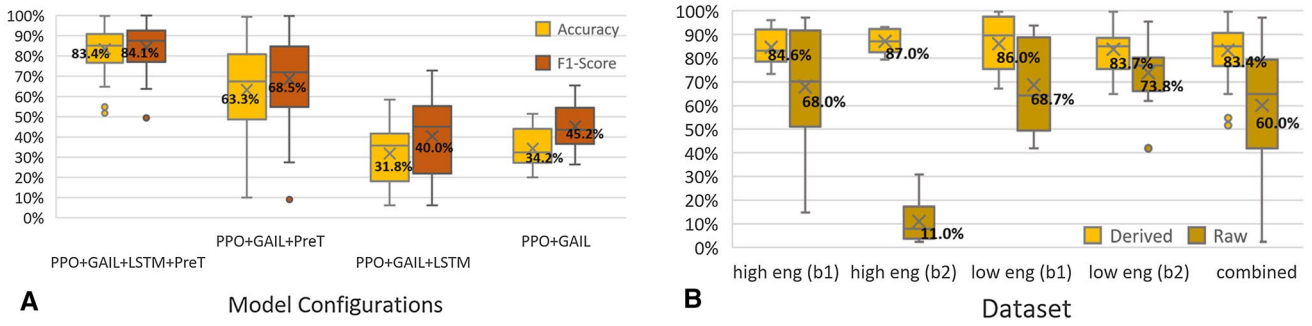

Fig. 4 **A** Accuracy and F1 score for all model configurations on 38 sessions. *PreT* stands for pre-training. **B** Accuracy values for PPO+GAIL+LSTM+PreTrain model configurations trained with derived and raw input data on 38 sessions. The *high eng* and *low eng* refers to high, respectively, low engagement data based on the human's annotations. The *(b1)* and *(b2)* represents the first or second batch in which the data were recorded. Finally, *combined* refers to the dataset that puts together all the high, low and average engagement data. The average engagement data are omitted as there are only 2 sessions, one in each batch. The accuracy for these are: from batch 1, 54.9% and 58.0% for the model with derived, respectively, raw data; from batch 2, 51.7% and 37.0% for the model with derived, respectively, raw data

Table 5 Confusion matrices for each model configuration. The confusion matrix for each configuration is an averaged confusion matrix from all 38 sessions. The *Low, Mix, High* are the categories, denoting high social engagement, mix social engagement and low social engagement. The rows show the actual (*Act.*) data (from the ground truth), the columns show the predicted (*Pred.*) data (the model's outcome)

	Pred.			
	Act.	Low	Mix	High
PPO+GAIL+**LSTM +PreT**	Low	145	2	21
	Mix	2	0	2
	High	21	1	97
PPO+GAIL+**PreT**	Low	136	19	14
	Mix	2	0	1
	High	32	41	47
PPO+GAIL+**LSTM**	Low	77	65	26
	Mix	2	1	0
	High	60	43	14
PPO+GAIL	Low	44	61	63
	Mix	1	1	1
	High	25	40	54

The algorithms in bold are additions to the base mode

does not have a temporal memory (LSTM). Although its average accuracy and F1-score are considerably higher than the other two configurations, the model results show a high variance compared to the best performing one (PPO+GAIL+LSTM+PreTrain), see Fig. 4A. Without considering the outliers, it registers values as low as 9.9% for F1-score and 27.4% for accuracy.

The remaining two models show a low performance: 31.8% accuracy, 40% F1-score for PPO+GAIL+LSTM configuration and 34.2% accuracy, 45.2% F1-score for the PPO+GAIL configuration. This indicates that pre-training has a significant contribution to the model configuration. However, pre-training and LSTM together with PPO and GAIL performs the best across all tested data.

The confusion matrices for all configurations are shown in Table 5. The three categories (*Low*, *Mix* and *High*) are a result of the second data post-processing (see Eq. 6). Unlike PPO+GAIL+LSTM+PreTrain, model configurations PPO+GAIL+PreTrain, PPO+GAIL+LSTM and PPO+GAIL have a high values in the *Mix* category: 60, 110 and 102 compared to the actual amount of the *Mix* category: 3. The *Mix* category represents roughly equal amounts of high and low social engagement values (1 and −1). High proportions of *Mix* is therefore likely to indicate a noisy model, the prediction fluctuating from one social engagement rating to another.

6.3 Derived versus raw features

Based on results in deep representation learning (Bengio et al. 2013), we hypothesise that the model trained with raw input data might yield similar results as the models trained with the derived data. The raw features are the base of the derived features. Therefore, an ML model with a complex configuration such as (PPO+GAIL+LSTM+PreTrain), which performed best with derived data, could be able to infer from the raw data and generalise to detect the engagement level in a social interaction (Bengio et al. 2013).

Therefore, we train the best performing configuration with raw data, following the same procedure to calculate the accuracy and F1-score. The mean values of these metrics are not too low, with 60% accuracy and 63% F1-score; however, there is a very high variance in the model's predictions (Fig. 4B -combined dataset). We collected the data used for training both types of models (with raw and derived data) in two slightly different setups (see

Table 6 Confusion matrices for the model configuration PPO+GAIL+LSTM+PreTrain, with derived and raw data. The confusion matrix for each model is an averaged confusion matrix separated in two data collection batches: first batch (with 18 sessions) and second batch (with 20 sessions). The *Low, Mix, High* are the categories, denoting *High*, *Mix* and *Low* social engagement. The rows show the actual (*Act.*) data (the ground truth), and the columns shows the predicted (*Pred.*) data (the model's outcome)

		Derived data			Raw data		
	Pred.	Low	Mix	High	Low	Mix	High
	Act.						
1st batch	Low	137	2	15	112	20	21
	Mix	2	0	2	2	1	1
	High	26	1	103	30	19	82
2nd batch	Low	152	3	27	**148**	**26**	**8**
	Mix	1	0	2	**2**	**1**	**0**
	High	16	1	90	**82**	**18**	**6**

Bold is the confusion matrix showing the low performance of the model trained with raw data on 2nd batch data

Sect. 4.2). Briefly, the first setup (batch 1) has the VC in a different location than in the second setup (batch 2); apart from that, the VC's gaze behaviour is triggered at objects at the exact same time in both batches; however, the VC is gazing at different objects in batch 1 compared to batch 2.

We suspected that the VC's new position (in batch 2) might have influenced the model trained with raw data. This is because the model performs well on the sessions from batch 1 (for both high and low social engagement), but very low on the sessions from batch 2, especially when trying to detect high social engagement. The difference between the two batches is in the VC's location. Since the input for training the model includes the VC's location, we consider this a potential reason.

To test this, we separate the results into each of two batches and into the engagement categories (low and high). Figure 4B shows a comparison of the two models' accuracy: one model trained on derived data and the other on raw data. The F1-score values have a very similar trajectory, hence they are omitted from the figure to not clutter it (see Fig. S5 in the supplementary documents for the F1-score values). Figure 4B explains the reason why the raw model has such a large accuracy (and F1-score) variance over these 38 sessions. The high engagement data from the second data recording batch register very low accuracy and F1-score values compared to the high engagement data from the first batch. There is no significant difference between the low engagement data from the first and second batch.

We ran a 2×3 Mixed ANOVA analysis (within-group factor *treatment*: raw input data, derived input data; between-group factor *VR stage*: *S*1, *S*2, *S*3). This reveals that the derived data performed significantly better than raw ($p < 0.001$), and that there is a significant *VR stage* effect($p = 0.045$), but no interaction effect was found ($p = 0.172$). Post hoc Tukey test reveals that the model performed significantly better for VR stage *S*3, as compared to VR stage *S*2($p = 0.035$). No other effects were found between VR stages.

Table 6 contains the confusion matrices for models significant effect on the accuracy and trained on derived and raw data split based on the data collection batch. The model trained on raw data fails to detect a large proportion of the *High* social engagement parts, mostly miss-predicting them as low social engagement. This model also shows a much higher fluctuation of social engagement rating per 1-s window. This is illustrated in the high amount of predictions for the *Mix* social engagement, 40 ($20 + 1 + 19$ in Batch 1) and 45 ($26 + 1 + 18$ in Batch 2) compared to the actual value of 4 ($2 + 1 + 1$ in Batch 1) and 3 ($2 + 1 + 0$ in Batch 2).

The model trained with raw data might have learned very specific features, for example, the exact position of the VC. If that condition is not fulfilled (the VC is not positioned on the same location or has a changing position), then the raw data model incorrectly predicts the engagement level. This is a problem as it is very common in games to have VCs that would move in the environment.

There could be a possible solution to improve the raw model's performance while keeping the VC active in the scene. To do this, more data needs to be collected with the VC in different locations and using more participants to interact with the VC. This might decrease the variance in accuracy for the raw model. However, the process of recording the data and training the model is very expensive and time-consuming, making it unfeasible for a game production process. The use of psychologically inspired derived features is therefore a better approach within the practical time and budget constraints of game development.

7 Generalisation: ML pipeline for social attitude detection

In this section we go over the pipeline used to detect social engagement (Sect. 4), generalising it to be used for detecting other social attitudes detection (such as sympathy, affection or aggression).

Both the data collection and the data annotation take place in VR. First, a user interacts with the VC and their behavioural data is collected (See Fig. 1A). Next (Fig. 1B), a human annotator labels the presence of a social attitude, while watching a playback of the user–VC interaction in VR.

The data from the user–VC interaction are the base data for training the ML model. It consists of data about the user's and VC's activity, such as the movement (head and hands position and rotation), interaction with other objects or with each other and so on. By performing this data collection in VR, we are able to create a situation that is as close as possible to real gameplays and also to real social interactions. The data collected in this step can be different to the one we collected for social engagement detection (Sect. 4); it should contain relevant data for the specific social attitude (e.g. eye or pulse information).

These data are then played back in VR to be labelled for training the ML model. Because of this, it should contain instances of the social attitude's presence (positive value) and its absence (negative value). Thus, the VR scenario needs to contain situations that allow both positive and negative examples of a participants' social attitude.

The human annotator is a key figure in this pipeline. They have the ability to look at the behaviours from the interaction and choose the ones that resemble the complex social attitude the ML model will learn to detect. In game development, a creative director could be the annotator. They use their artistic vision on the final product and decide what social attitude is important to be detected in a particular scene in the game, while a user is socially interacting with a VC/NPC.

To decide this, instead of having to provide a concrete definition of the social attitude, the annotator labels it while observing the playback interaction.

In our social engagement example, they can perform this action by pressing a 'plus' button on the VR controller when they see the attitude (social engagement) and a 'minus' button when there is the lack of it (Fig. 3B). This way, the annotations conceptualise the complex and abstract activity (social attitude). Then, the trained ML model will detect this activity during social interaction. Other features could be included in the labelling task in VR to ease and improve the outcome based on the social attitude (e.g. video feed overlay).

By annotating in VR, the annotator is able to make full use of their social cognition skills as they would do in a real-world social interaction. By performing it in a game environment, it can become a game design task, in which a designer can judge the interaction as it would fit into the real gameplay.

The ML model trains on the dataset: the user's data and the annotator's labels as ground truth data (Fig. 1C). The ML model replicates the human's annotations by using an imitation learning algorithm approach, thus mimicking the human intuition of marking an attitude within a social interaction (Fig. 1D).

After training the model for detecting, for example, aggression, it can be applied to different scenarios. The model outputs whether aggression is present or absent based on the input data from the interaction (Fig. 1E–G). Finally, the output can be manipulated and used real time in applications (e.g. games) to trigger various actions or behaviours based on the designer's vision. For instance, when the player is detected to be too aggressive, the NPC could stop talking being animated to reflect the behaviour received; when the player is showing empathy, the NPC will start talking again and their animation would change indicating that.

8 Limitations and discussion

The exact results on the social engagement detection presented in the paper could be difficult to replicate without the same annotator. However, the aim of the project was not to create a general detection model for social engagement (or other social attitudes) because individuals often have their own standards of what counts as engaged or not (Glas and Pelachaud 2015). In our case study of detecting social engagement using the proposed pipeline, instead of explicitly defining the present or absent criteria of a certain social attitude, we rely on the annotator's ability to label it. In game companies, this annotator role should be taken by their creative director, making the labelling itself part of the creative process. In other words, we aim to detect the High/Low social engagements that are modelled based on the creative game designer (the annotator) markings. During the VR interaction playback, they would be labelling the behaviours identified in players which are related to the presence or absence of social engagement. Thus when real-life players exhibited those behaviours during gameplay, certain events (NPC behaviours, or change of game environment) could be then triggered.

For this work, the annotator marked the data in a binary way: either high or low social engagement data. This can be a limitation to our approach, since social attitudes are not necessary binary. We decided to go with this approach because the algorithms within the Unity ML agents have

the requirement of a binary input data hence we kept the annotations and the model's predictions binary. However, the model predicts at a higher frequency (9 to 10 frames per second) compared to other work such as Yu et al. (2004) where predictions take place per each utterance, or Bohus and Horvitz (2014) where they use a 5-s window to forecast disengagement. In this case, the game system can use the prediction model at finer level. For instance, the game designer can choose to calculate an averaged social engagement over a time spam of 5 s (as in Bohus and Horvitz 2014). For that time window there will be a total of 45–50 ($5 \times 9, 5 \times 10$) predictions which can be used to calculate a fine-grained outcome, rather than a binary one.

We ensured the annotator was making judgements only on the data available to the ML model. If the annotator was making judgement using data inaccessible to the ML algorithm, we argue that the algorithm cannot learn by it, as described in Gillies et al. (2015). However, we based this decision on prior literature and we did not attempt to annotate the interactions on more data compared to the one used for training the models. For future work, we could rerun the annotation process on the user–VC interactions giving the annotator more data. Then we could train the ML models using these labels but with less data than the one used for labelling. We could then compare the results with the ones already reported in this paper.

We collected data from participants in a western city who volunteered to take part in the study hence they might have a interest in XR. For this reason, the behaviour and social attitude expression recorded are linked to the cultural background. As future work we plan to run studies with participants from other backgrounds to enrich and compare the dataset and the detection model. Even though we collected data from 38 sessions and from 13 participants, the dataset was not very large. We also selected features that were readily available to train the ML model. It would be an interesting future work to consider different array of features available from more cutting edge hardware.

We run the data collection in two batches to investigate the effect of the agents' gaze at various key objects. This aspect is out of scope for this paper; however, recording the data in two batches with a distinct VC location for each allowed for a more diverse dataset (see Sect. 4.2). As future work, we could diversify it even more by assigning a different VC starting point for each participant.

Despite our limitations, we received very positive comments from our industry collaborators. The industry collaborators helped to create the tool, and as a retrospective note, they commented on how non-verbal communication being in the centre of the tool, and that the use of non-verbal behaviour widens the applicability of this work for other types of games and application within the entertainment and games industries. During the collaboration we chose the social engagement case study; however, one of the game companies applied the pipeline for developing a VR karaoke-style application where the user would sing along an NPC singer, which will change their attitude depends on the social attitude of the player in real time. Likewise, they did not define the social attitude, but the annotation was based on the way the players sang, moved, performed, and how involved they were in the experience. The CEO of the game company commented on their experience of building the VR karaoke app using the immersive ML pipeline: 'The ML project was very interesting to be a part of, seeing it grow from a very simple idea into something quite sophisticated. What impressed me the most was seeing the same principles used in a nineteenth century narrative game also applied to a modern karaoke game. This generalisation convinced me of the merit of the approach taken. From experience I think it is relatively straight forward to get a system working on one context, but to reapply the same principles in a fundamentally different context proves it's true worth.'

9 Conclusion

In this paper, we present our collaborative work with two games companies to develop a pipeline with an immersive data collection and annotation in VR for training an ML model. We design the pipeline to support the games industry creative design process and to be integrated into production-ready VR games for the consumer market.

The pipeline is used to train a ML model to detect social engagement using a reinforcement learning (PPO) approach with rewards based on an imitation learning algorithm (GAIL). We also presented the pipeline as a general tool to detect social attitudes, such as sympathy or aggression.

We consider different model configurations and input data for training the model: derived data and raw data. The model using derived data performs the best, while the model based on raw data is not able to generalise to different VC positions. The model configuration that yields the highest accuracy and F1-score (83.4%, 84.1%) is based on a reinforcement learning algorithm (PPO) with imitation learning rewards (GAIL) implementing a temporal memory (through LSTM) and a pre-training algorithm. The other model configurations perform poorly, the outcome being a rapid change between high and low social engagement values in a short period of time.

The proposed work contributes to the field of socially responsive VCs, offering a design by example tool for immersive ML, to detect social engagement (and possibly abstract social attitudes) in VR social interactions. This could be useful in designing social interactions in VR games or in other immersive experiences (simulations, training, social platforms), where the user can interact with

the VC using their own bodies, as they do in everyday life. This opens opportunities for novel input interactions, game mechanics or VC's behavioural models that are related to the rapport/empathy between the user(s) and the VC.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10055-022-00644-4.

Funding All participants took part in the experiment voluntarily and signed a consent form. They were given the freedom to drop at any time. The project was approved by the University's ethics board. We worked on this project in collaboration with two game studios: *Dream Reality Interactive* (https://www.dreamrealityinteractive.com/) and *Maze Theory* (https://www.maze-theory.com/); the project being supported by *Innovate UK* Grant TS/S02221X/1. This work was also partly supported by Grant EP/L015846/1 for the Centre for Doctoral Training in Intelligent Games and Game Intelligence (http://www.iggi.org.uk/) from the UK Engineering and Physical Sciences Research Council (EPSRC).

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Ahuja C, Ma S, Morency LP, Sheikh Y (2019) To react or not to react: end-to-end visual pose forecasting for personalized avatar during dyadic conversations. In: 2019 International conference on multimodal interaction, pp 74–84

Bailenson J (2018) If a possible mass shooter wants to hone his craft, don't hand him a virtual boot camp. https://edition.cnn.com/2018/03/05/opinions/video-games-shooting-opinion-bailenson/index.html, https://edition.cnn.com/2018/03/05/opinions/video-games-shooting-opinion-bailenson/index.html

Bee N, Franke S, Andreé E (2009) Relations between facial display, eye gaze and head tilt: dominance perception variations of virtual agents. In: 2009 3rd international conference on affective computing and intelligent interaction and workshops. IEEE, pp 1–7

Bengio Y, Courville A, Vincent P (2013) Representation learning: a review and new perspectives. IEEE Trans Pattern Anal Mach Intell 35(8):1798–1828

Bohus D, Horvitz E (2014) Managing human–robot engagement with forecasts and... um... hesitations. In: Proceedings of the 16th international conference on multimodal interaction, association for computing machinery, New York, NY, USA, ICMI'14, pp 2–9, https://doi.org/10.1145/2663204.2663241

Brugel S, Postma-Nilsenová M, Tates K (2015) The link between perception of clinical empathy and nonverbal behavior: The effect of a doctor's gaze and body orientation. Patient Educ Counsel 98(10):1260–1265. https://doi.org/10.1016/j.pec.2015.08.007 communication in Healthcare: Best papers from the International Conference on Communication in Healthcare, Amsterdam, The Netherlands, 28 September–1 October 2014

Burgoon J, Dillman L, Stem L (2006) Adaptation in dyadic interaction: defining and operationalizing patterns of reciprocity and compensation. Commun Theory 3:295–316. https://doi.org/10.1111/j.1468-2885.1993.tb00076.x

Cafaro A, Ravenet B, Ochs M, Vilhjálmsson HH, Pelachaud C (2016) The effects of interpersonal attitude of a group of agents on user's presence and proxemics behavior. ACM Trans Interact Intell Syst 6(2):2914796. https://doi.org/10.1145/2914796

Chinchor N (1992) Muc-4 evaluation metrics. In: Proceedings of the 4th conference on message understanding, association for computational linguistics, USA, MUC4'92, pp 22–29, https://doi.org/10.3115/1072064.1072067

Christensen JV, Mathiesen M, Poulsen JH, Ustrup EE, Kraus M (2018) Player experience in a vr and non-vr multiplayer game. In: Proceedings of the virtual reality international conference-Laval virtual, pp 1–4

Dermouche S, Pelachaud C (2019a) Engagement modeling in dyadic interaction. In: 2019 international conference on multimodal interaction, pp 440–445

Dermouche S, Pelachaud C (2019b) Generative model of agent's behaviors in human-agent interaction. In: 2019 international conference on multimodal interaction, pp 375–384

Dhamija S, Boult TE (2017) Automated mood-aware engagement prediction. In: 2017 seventh international conference on affective computing and intelligent interaction (ACII). IEEE, pp 1–8

Feng W, Kannan A, Gkioxari G, Zitnick CL (2017) Learn2smile: learning non-verbal interaction through observation. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), pp 4131–4138

Ferstl Y, McDonnell R (2018) Investigating the use of recurrent motion modelling for speech gesture generation. In: Proceedings of the 18th international conference on intelligent virtual agents. ACM, pp 93–98

Forbes-Riley K, Litman D, Friedberg H, Drummond J (2012) Intrinsic and extrinsic evaluation of an automatic user disengagement detector for an uncertainty-adaptive spoken dialogue system. In: Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: human language technologies, Association for Computational Linguistics, Montréal, Canada, pp 91–102, https://www.aclweb.org/anthology/N12-1010

Gillies M, Kleinsmith A, Brenton H (2015) Applying the CASSM framework to improving end user debugging of interactive machine learning. In: International conference on intelligent user interfaces, proceedings IUI, vol 2015, https://doi.org/10.1145/2678025.2701373

Glas N, Pelachaud C (2015) Definitions of engagement in human-agent interaction. In: 2015 international conference on affective computing and intelligent interaction (ACII), IEEE, pp 944–949

Gordon G, Spaulding S, Westlund JK, Lee JJ, Plummer L, Martinez M, Das M, Breazeal C (2016) Affective personalization of a social robot tutor for children's second language skills. In: Thirtieth AAAI conference on artificial intelligence

Greenwood D, Laycock S, Matthews I (2017) Predicting head pose in dyadic conversation. In: International conference on intelligent virtual agents. Springer, pp 160–169

Hale J, Ward JA, Buccheri F, Oliver D, Hamilton AFdC (2020) Are you on my wavelength? Interpersonal coordination in dyadic conversations. J Nonverbal Behav 44(1):63–83

Hall ET (1966) The hidden dimension, vol 609. Doubleday, Garden City

Ho J, Ermon S (2016) Generative adversarial imitation learning. In: Advances in neural information processing systems, pp 4565–4573

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

Ip B (2011) Narrative structures in computer and video games: Part 1: context, definitions, and initial findings. Games Cult 6(2):103–134

Jin A, Deng Q, Zhang Y, Deng Z (2019) A deep learning-based model for head and eye motion generation in three-party conversations. Proc ACM Comput Graph Interact Tech 2(2):1–19

Khaki H, Bozkurt E, Erzin E (2016) Agreement and disagreement classification of dyadic interactions using vocal and gestural cues. In: 2016 IEEE international conference on acoustics. Speech and signal processing (ICASSP). IEEE, pp 2762–2766

Lee SP, Badler JB, Badler NI (2002) Eyes alive. In: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, pp 637–644

Marsella S, Xu Y, Lhommet M, Feng A, Scherer S, Shapiro A (2013) Virtual character performance from speech. In: Proceedings of the 12th ACM SIGGRAPH/Eurographics symposium on computer animation, pp 25–35

Max Roser CA, Ritchie H (2013) Human height. Our World in Data https://ourworldindata.org/human-height

Mota S, Picard RW (2003) Automated posture analysis for detecting learner's interest level. In: 2003 Conference on computer vision and pattern recognition workshop, vol 5, pp 49–49

Pan X, Hamilton AFdC (2018) Why and how to use virtual reality to study human social interaction: the challenges of exploring a new research landscape. Br J Psychol 109(3):395–417

Pan X, Collingwoode-Williams T, Antley A, Brenton H, Congdon B, Drewett O, Gillies MFP, Swapp D, Pleasence P, Fertleman C et al (2018) A study of professional awareness using immersive virtual reality: the responses of general practitioners to child safeguarding concerns. Front Robot AI 5:80

Sanghvi J, Castellano G, Leite I, Pereira A, McOwan PW, Paiva A (2011) Automatic analysis of affective postures and body motion to detect engagement with a game companion. In: Proceedings of the 6th international conference on human–robot interaction, association for computing machinery, New York, NY, USA, HRI'11, pp 305–312, https://doi.org/10.1145/1957656.1957781

Schilbach L, Timmermans B, Reddy V, Costall A, Bente G, Schlicht T, Vogeley K (2013) Toward a second-person neuroscience. Behav Brain Sci 36:393–414. https://doi.org/10.1017/S0140525X12000660

Schmidt A (2000) Implicit human computer interaction through context. Pers Technol 4(2):191–199

Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347

Shao K, Tang Z, Zhu Y, Li N, Zhao D (2019) A survey of deep reinforcement learning in video games. arXiv:1912.10944

Slater M (2009) Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments. Philos Trans R Soc B: Biol Sci 364(1535):3549–3557

Slater M, Steed A (2000) A virtual presence counter. Presence Teleoper Virtual Environ 9(5):413–434

Steed A, Schroeder R (2015) Collaboration in immersive and non-immersive virtual environments. In: Immersed in media. Springer, pp 263–282

Vinciarelli A, Pantic M, Heylen D, Pelachaud C, Poggi I, D'Errico F, Schroeder M (2011) Bridging the gap between social animal and unsocial machine: a survey of social signal processing. IEEE Trans Affect Comput 3(1):69–87

Wilson G, McGill M (2018) Violent video games in virtual reality: re-evaluating the impact and rating of interactive experiences. In: Proceedings of the 2018 annual symposium on computer–human interaction in Play, pp 535–548

Woolf B, Burleson W, Arroyo I, Dragon T, Cooper D, Picard R (2009) Affect-aware tutors: recognising and responding to student affect. Int J Learn Technol 4(3–4):129–164

Yu C, Aoki PM, Woodruff A (2004) Detecting user engagement in everyday conversations. arXiv preprint arXiv:cs/0410027