A. Social agents are not objects.

Although social peers could be seen as merely complex interactive objects, we argue they are in essence quite different. Social agents (e.g. humans) can have very complex and changing internal states, including intents, moods, knowledge states, preferences, emotions, etc. The resulting set of possible interactions with peers (social affordances) is essentially different from those with objects (classical affordances). In cognitive science, an affordance refers to what things or events in the environment afford to an organism (Carvalho, 2020). A flat surface can afford "walking-on" to an agent, while a peer can afford "getting help from". The latter is a social affordance, which may require a social system and conventions (e.g. politeness), implying that social peers have complex internal states and the ability to reciprocate. Successful interaction might also be conditioned on the peer's mood, requiring communication adjustments. Training an agent for such social interactions most likely requires drastically different methods – e.g. different architectural biases – than classical object-manipulation training. In SocialAI we simulate such social peers using scripted peers. We argue that studying isolated social scenarios featuring scripted peers in tractable environments is a promising first step towards designing proficient social agents.

B. Extended Cognitive Science Background

The following section introduces additional detail to section 3.

B.1. Michael Tomasello - The Shared Intentionality Theory

Discussion on the uniquely human socio-cognitive abilities Michael Tomasello (2019) argues that the 9-month-revolution and the objective/normative turn are uniquely *human* developmental steps enabling uniquely *human* socio-cognitive abilities. There has been a lot of debate regarding this hypothesis (De Waal, 2016), and it still remains an open question. However, for the purpose of this work, the social proficiency of other great apes (or our last common ancestor with them) is not of primary importance. We find The Shared Intentionality Theory useful because it is systematic, extensive (covers a broad range of social abilities), and exact (is build upon a number of very clearly defined experiments). Furthermore, it is concerned with the questions regarding the development of core socio-cognitive abilities. We believe that this makes it a good basis to organize AI research on.

B.1.1. Social cognition

Imagining what others perceive In this paragraph, we discuss the ability to infer what another sees or knows. As compared to the later emerging ability to coordinate perspectives, this ability requires that only one perspective is processed at a time. Numerous studies have shown that both apes and children are capable of making such inferences. For example, in one experiment (Hare et al., 2001), a subordinate and a dominant chimpanzee were presented with a competitive scenario : competing for food. Results showed that the subordinate chimpanzee attempted to eat the food only if it was hidden from the dominant one. This experiment was then extended to children who were presented with two toys: one observed by an adult and one occluded from him (Moll & Tomasello, 2006). When asked to help the adult find a toy, 24-month-olds passed the occluded toy. These experiments, demonstrate that both children and apes are capable of inferring what a conspecific observes - i.e. they are able to infer another's perspective.

Joint Attention Joint attention is often defined in different ways (Siposova & Carpenter, 2019). To avoid confusion, we take the definition of joint attention from Michael Tomasello (Tomasello, 2019): joint attention consists of two aspects: *triangulation* and *recursiveness*. Triangulation refers to the child and the adult attending to the same external referent, and recursiveness refers to them both recursively being aware that they are both sharing attention. Joint attention enables children to process multiple perspectives at the same time, which can then be aligned and exchanged.

In a series of studies (Carpenter et al., 1998b), the amount of joint attention (number of joint attention episodes and their length) was measured in free play interactions between infants and their mothers. A steady rise in the amount of time spent in joint attention was observed in the period from 9 to 12 months.

Coordinating perspectives When sufficient linguistic competence is reached, children start to jointly attend to mental content through linguistic discourse. Through linguistic discourse children often encounter conflicting perspectives, which they are then pushed to resolve (e.g. one parent says it's raining outside, but another says it's not). They resolve those conflicts by learning to form an "objective" perspective - a bird's-eye-view perspective distinct from anyone's personal perspective - and coordinating the conflicting perspectives with it. (Tomasello, 2019) argues that this can only be achieved

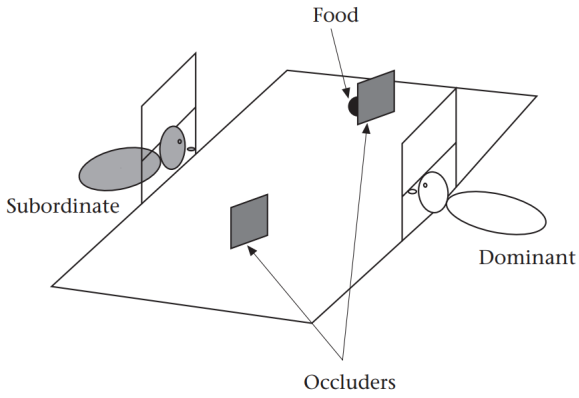


Figure 4. Sketch of an experiment showing that apes can infer the conspecific's field of view (Hare et al., 2001). As the subordinate ape does not want to get into trouble, it will not steal the food from the dominant ape. In the experiment, the food was either occluded from the dominant ape or placed in plain sight. The subordinate ape ate the food only when it was occluded from the dominant ape. This shows that it was able to infer the dominant's field of view.

once a child has passed through the second developmental step, that of collective intentionality, which enables them to form such a "perspectiveless" bird's-eye view perspective.

B.1.2. Communication

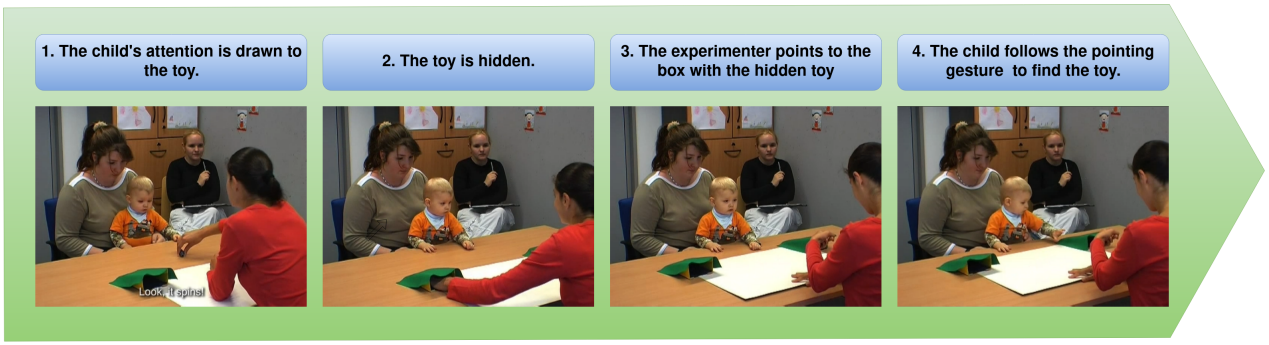


Figure 5. An experiment studying children's ability to infer the meaning of a pointing gesture (Behne et al., 2005). The child's attention is drawn to a toy. This toy is then hidden in one of the two boxes (the child does not know which one). The experimenter then points to one of the two boxes, and the child is able to infer this to mean that the toy is in that box.

Referential communication - The Pointing gesture Following the 9-month revolution, children start to communicate *referentially* - to an external referent (Tomasello, 2019). This is primarily achieved through *pointing* and pantomiming.

Figure 5 depicts an experiment studying the children's understanding of the pointing gesture (Behne et al., 2005). First, the child's attention is drawn to the toy, which is then hidden in one of the two boxes. The experimenter then points to a box, and the child infers this to mean that the toy is in that box. 14-month-old children were able to successfully follow a pointing gesture to find the toy. In this scenario, the child makes the following recursive inference: the adult is helping by directing the child's attention to the box, and she wants the child to infer that the toy is in the box.

Linguistic communication Linguistic communication is based on the same principle as gestural referential communication: sharing attention to a referent and recursively inferring the intended meaning. However, linguistic communication in addition requires learning conventionalized means of reference, such as words or phrases. Where once was a single pointing gesture,

now there is a complex grammar of gestures, with specific conventions assigned to each gesture. In a series of studies, Malinda Carpenter (1998b) observed children's understanding of words steadily increase in the period after 9 months. This was measured by questioners given to their caretakers on regular intervals.

Michael Tomasello argues that when language use first appears, children do not yet understand it as conventional, rather they use it as any other artifact or tool. It is only after the emergence of collective intentionality, when children start to understand and use conventions and norms, that they also begin to perceive language as such. This is evidenced by specific new ways in which they come to use and understand language. For example, when others break the rules of a game they protest by normative statements such as "No! It does not go like this!" (Wyman et al., 2009). It is needless to say that language plays many important roles in children's development. Here we will outline just a few of countless possible examples. Language provides children with abstract constructions which gives them a new organized format for cognitive representation. Through discourse, children encounter many conflicting perspectives, which brings them to resolve those conflicts by forming the "objective" perspective. Finally, language opens up a new way of cultural learning - instructed learning - in which adults directly teach children "objective" truths about the world. Knowledge learned in that manner is easier to generalize (Butler & Tomasello, 2016).

B.1.3. Cultural Learning

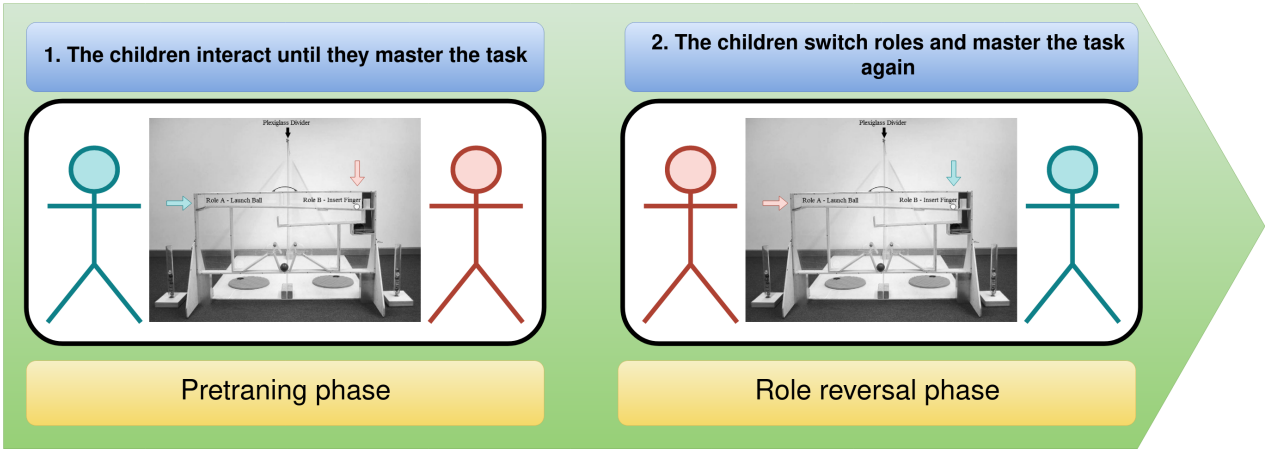


Figure 6. Depiction of an experiment on role reversal (Fletcher et al., 2012). The task consists of two roles: one participant pushes a ball into the apparatus, and the other redirects it with their finger. The ball then pushes two marbles toward each of the participants. In the pretraining phase, children collaborate until they master the task (three consecutive successful trials). Then, in the role reversal phase, their roles are reversed and they master the task again. Total number of trials to master the task is compared between the two phases. Children, but not apes, needed less trials to master the task in the role reversal phase than in the pretraining phase.

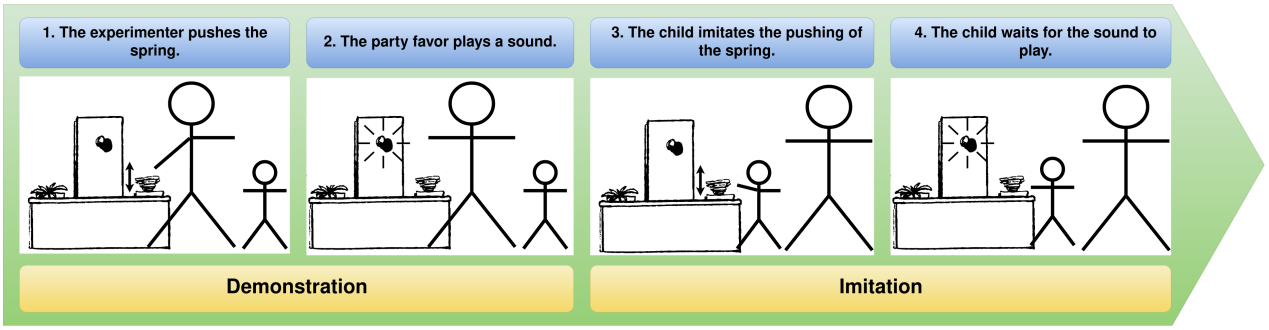


Figure 7. Depiction of an experiment(Carpenter et al., 1998b). The experimenter activates the party favor (sound) by pushing the spring, and the child imitates and waits for the sound. The sketch was taken and modified from Carpenter (1998b).

Imitation, Emulation, and Overimitation Imitation and emulation learning both refer to observing a demonstration and learning from it. Imitation learning refers to the learning of means (actions), while emulation to the learning of ends (goals) of a demonstration (Whiten et al., 2004; 2009; Tennie et al., 2006). Refer to Whiten (2009) for a discussion and taxonomy of imitative and emulative learning processes.

Figure 7 shows an experiment studying children's imitation abilities (Carpenter et al., 1998b). In this experiment, the experimenter demonstrates an instrumental action (e.g. pressing a spring attached to a box) which activates the light on top of the box. The children repeated the instrumental action and looked expectedly at the light. This kind of learning emerges over the course of the first year - children reconstruct the outcome of others' actions. However, soon after this, children begin imitating in a way which demonstrates the understanding of other's goals. Children perform an action that an adult attempted, but failed to perform (Meltzoff, 1995), and do not imitate accidental actions (Carpenter et al., 1998a). Similarly, rational imitation appears. If the action was forced upon the demonstrator, the children recreate the result through more rational means (Gergely et al., 2002). For example, in an experiment a demonstrator pressed a button with its head while having tied hands and 14-month-olds responded by pressing the button with their hand (Gergely et al., 2002).

Emulation is a type of social learning where the focus is on the outcome, and not on the actions performed (Wood et al., 1989). In other words, the learning is about some property of the environment rather than the social interaction. For example, by observing someone turn on the lights we learn that lights can be turned on by flicking a light switch. The learner tries to recreate some observed outcome, in doing so they can, but do not have to, recreate the actions.

On the other side of this spectrum is *overimitation* - children repeat actions that are not relevant for the outcome. Children often prefer to not only recreate the outcome (as in emulation), but also do it in the same way as the adults (even if this requires doing additional unnecessary actions). For example, in one experiment (Tennie et al., 2014) children were presented with a demonstration of a rice-pouring task. The experimenter performed a useless preliminary action before grabbing the rice. 4-year-old children responded by repeating both the useless and the necessary actions. It has been proposed that children overimitate to affiliate and conform for the purpose of in group bonding (Over & Carpenter, 2013), but this remains an open question (Keupp et al., 2013; Lyons et al., 2007)

Role reversal Imitation Role reversal imitation appears following the 9-month revolution. Figure 6 depicts an experiment with children and apes from Fletcher (2012). An apparatus is used where one participant pushes the marble, and the other inserts a finger to redirect the ball so that it falls to the correct location. Then, both participants get a reward. Children who previously played role A mastered role B in less trials than children who never played role A. In another experiment (Carpenter et al., 2005), children were asked to immediately reverse the role. An experimenter did some action on the child (e.g. poke the child and say "your turn") and the child responded with the same action on the experimenter (poke the experimenter back). These experiments show that children understand the separate roles and how each is relevant for the activity.

C. Technical details on the SocialAI school

In this section, we discuss in detail the parameterized environments and their procedural generation. Then, we discuss the architecture of the RL agent and the exploration bonuses used in our experiments.

C.1. Parameterized Social Environments

The SocialAI school is built on top of the MiniGrid codebase (Chevalier-Boisvert et al., 2018), which provides an efficient and easily extensible implementation of grid world environments. SocialAI environments are grid worlds consisting of a room. In all of our environments, the task of the agent is to eat the apple, at which point it is rewarded. The reward is diminished according to the number of steps it took the agent to complete the episode. The episode ends when the agent eats the apple, uses the *done* action, or after a timeout of 80 steps.

The agent interacts with the environment as shown in figure 8. This multimodal observation space consists of the full dialogue history, and a 7x7x8 tensor corresponding to the 7x7 grid in front of the agent. Each cell is encoded by six integers representing the object type, color, and some additional object-dependent information (e.g. is the door open, point direction, gaze direction, etc.). Refer to figure 22 for a list of all objects.

The agent acts in the environment through a multimodal action space, which consists of 6 primitive actions (*no*, movement actions, *toggle*, and *done*) and a 4x16 templated language. The agent also has the option not to speak, which is implemented

with an additional binary output from the agent. Refer to section C.4 for further details about the architecture of the agent.

All environments contain a scripted social peer, and the task can only be solved by interacting with this peer (for which socio-cognitive abilities are needed). A social peer observes the world in the same way as the agent does (as a grid in front of it), and it also observes the agent's utterances. Their action space consists of primitive actions for movement, pointing, and the *toggle* action. The peer can also communicate with words and sentences. As the peer is scripted, there are no constraints on the language it can utter (it is not constrained to a templated language). The language it uses depends on the environment, which defines which sentence the peer will utter at which point. The peer is represented in the agent's observation by 7 integers depicting their: object type, position, color, type (cooperative or competitive), gaze direction, point direction, and the last executed primitive action. The peer's gaze and point directions are represented relatively the agent (e.g. 1 - to the left of the agent). The pointing direction can also be set to 0, which signifies that the peer is not pointing. Figure 9 shows an example of an environment with the corresponding encoding of the peer. The agent (red) and the scripted peer (purple) are making eye contact - the peer and the agent are in the same row or column and their gazes meet frontally. In this example, the scripted peer is also pointing to the blue box.

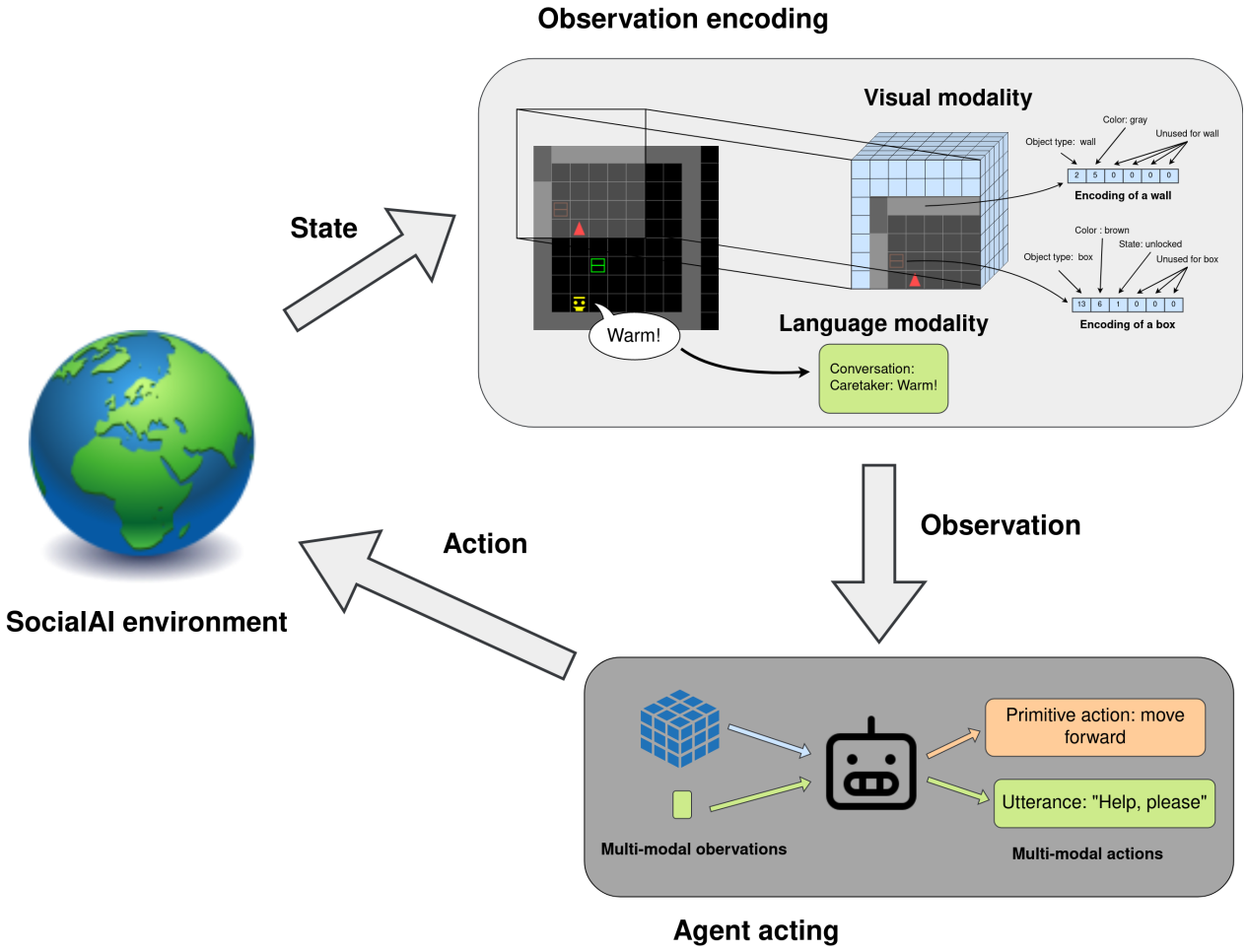


Figure 8. Workflow of an agent acting in the SocialAI school. The environment generates a state, which is represented as multimodal observations: a 7x7x6 tensor and the full dialogue history. The agent acts through a multi-modal action space consisting of primitive actions and utterances.

The SocialAI environments are parameterized, and those parameters define the social dimensions of the task. In other words, parameters define which socio-cognitive abilities are needed to solve the task. For example, depending on the ENVIRONMENT

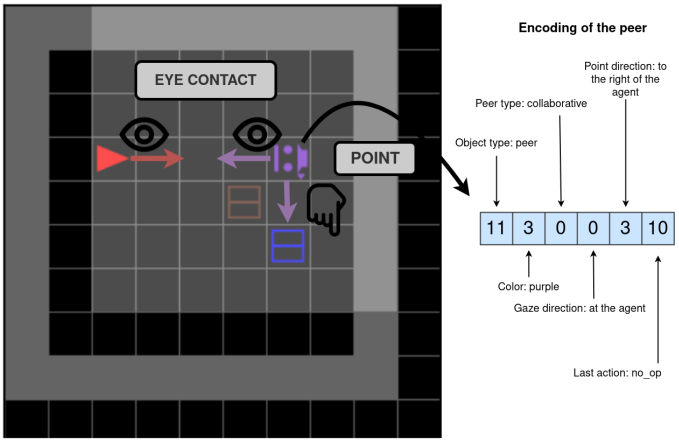


Figure 9. A depiction of a peer and its encoding. The agent and a peer are in eye contact, and the peer is pointing to the blue box. To the right is an encoding of the peer. The encoding contains information about the peer, e.g. the gaze and point direction.

TYPE parameter, the peer can give information, collaborate with the agent, or be adversarial. Then, in the case of the peer giving information, additional parameters define what is the form of this information (linguistic or pointing).

C.2. The parameter tree

SocialAI enables the creation of many parameterized environments, and those parameters are implemented as nodes in a parameter tree. A parameter tree is a structure through which the experimenter can easily define which parameters (and their values) can be sampled. An example of such a tree can be seen in figure 10. The standard procedure is that an experimenter defines such a parameter tree and each episode begins with the sampling of a new parameter set from this tree. Once a parameter set has been sampled, an environment is created, and the agent placed inside.

A parameter tree is used to sample parameter sets from it, three examples of such sampling are shown in figure 10. There are two kinds of nodes: parameter nodes (rectangles) and value nodes (ovals). Parameter nodes correspond to parameters, and value nodes corresponds to possible values for those parameters. Sampling proceeds in a top-down fashion, starting from the root node. In all our experiments, the root node is the ENV_TYPE parameter node. Sampling from a parameter node selects one of its children (a value node), i.e. sets a value for this parameter. This can be done by uniform sampling over the node's children, or by prioritized sampling with a curriculum. Once a value node has been chosen, the sampling continues through it to all of its children (parameter nodes). In other words, setting a value for one parameter, defines which other parameters (the value node's children) need to be set. In our codebase, it is simple to create such trees, and to add additional parameters and environments. In the remained of this section, we explain some of the most relevant parameters.

C.3. Environment types

The most important parameter in the parameter tree is the environment type (ENV_TYPE). This parameter node is the root node in all our experiments. We implemented three different values: INFORMATIONSEEKING, COLLABORATION, and ADVERSARIALPEER. A parameter tree doesn't have to contain all of those values, this choice entirely depends on the type of experiment one wants to conduct, most often only one of type will be present in a tree. For example, figure 10 shows the tree with only the INFORMATIONSEEKING environment type. This tree was used to study understanding of the pointing gesture in section 5.1.

Information Seeking type

We used this environment type in experiments studying communication, joint attention, and imitation learning. In figure 11 we can see examples of INFORMATIONSEEKING type environments.

The general principle of this environment type is as follows. The agent is rewarded upon eating the apple, which is hidden at the beginning of the episode and can be accessed through interacting with an object.

The PROBLEM parameter defines which objects will be present in the environment, and different objects make the apple

accessible in different ways (ex. pushing, toggling, ...). We implement six different problems: BOXES, SWITCHES, MARBLE, GENERATORS, DOORS, or LEVERS. For example, opening the box will make the apple appear at the location of the box, while pulling the lever will open the door in front of the apple.

The N parameter defines the presence of a distractor (present if set to 2) A distractor is an object of the same type as the correct object, but if used both objects are blocked and the apple cannot be obtained in this episode.

To find out which object is the correct one, the agent must interact with the scripted peer. This interaction starts with the agent introducing itself. The way in which the agent should introduce itself is defined by the INTRODUCTORY SEQUENCE parameter. We define the following four values: NO, EYE_CONTACT, ASK, ASK-EYE_CONTACT. For the value NO, no introduction is needed. The peer will give information at the beginning of the episode. In most of our experiments, we will use the value EYE_CONTACT where the peer turns to look at the agent and waits for the agent to look back (to establish eye contact) - the agent must direct its gaze directly towards the scripted peer. An example of an established eye contact can be seen in figure 9. For the value ASK, the agent needs to utter "Help, please". The agent does so using templated language, by selecting the "Help, *X*" template and the word "please". A full grammar of the language is given in table 2 in the appendix. Finally, the ASK-EYE_CONTACT value is a combination of the previous two. It requires that the agent utters "Help, please" during eye contact.

Once the agent introduces itself, the HELP parameter defines the peer's behavior. If it is set to Y the peer with obtain the apple, and leave it for the agent to eat. Alternatively, it will give cues to the agent about which object to use. The nature of this cue is defined by the CUE TYPE parameter. We define four different values: POINTING, LANGUAGE COLOR, LANGUAGE FEEDBACK, and IMITATION. For the POINTING type, the peer will point to the correct object. It will move to a location from which it can unambiguously point (e.g. the same row) and point to the object (see figure 11(a)). For the LANGUAGE COLOR type, the peer will say the color of the correct object (see figure 11(b). For the LANGUAGE FEEDBACK type, the peer will hint to the proximity of the agent to the correct object. Every step, the peer will utter "Cold", "Medium", "Warm" or "Hot, depending on how close the agent is to the correct object. For example, "Cold" means that the agent is far from the object, and "Hot" that it is right next to it (see figure 11(c). For the IMITATION type, the peer will demonstrate the use of the correct object. The peer will use the correct object, obtain the apple, and eat it, and then reset the environment to its initial state.

For the purpose of analyzing the agent's behavior in more detail, Information seeking environments can also be created without the peer (by setting the PEER parameter to N). The asocial version refers to an environment contains no distractor and no peer, i.e. the agent just needs to use the object in the environment.

Collaboration type

We used this environment type to study the ability of the agent to reverse the roles of a collaborative activity. Environments are separated into two halves (corresponding to different roles) by a fence over which the agent can see, but which it cannot cross (see figure 12). If both roles are fulfilled correctly, two apples will become accessible (one on each side of the fence).

The most important parameters are ROLE and PROBLEM. The parameter ROLE designates the role to be assigned to the agent. The PROBLEM parameter defines the collaborative activity, of which we implemented seven: DOORLEVER, MARBLEPUSH, MARBLEPASS, BOXES, SWITCHES, GENERATORS, MARBLE. In DOORLEVER one participant opens the door by pulling the lever and the other passes through them, and activates the generator (generating two apples). In MARBLEPUSH one participant opens the door by pulling the lever, and the other pushes a marble through them. This marble activates the *marble generator* upon contact with it. In MARBLEPASS one participant pushed the marble to the right side of the room, and then the other pushes it towards the *marble generator*. In the remaining four problems, one participant is presented with two boxes of different colors. The other participant is presented with two objects of the same colors as the two boxes and of the type defined by the PROBLEM parameter (e.g. two generators). First, the participant that was presented with boxes open one box (an apple will be in both). After this, to obtain its apple, the other participant must use the object of the same color as the opened box. In figure 12 we can see examples of COLLABORATION type environments.

Like the information seeking environments, collaboration environments can also be instantiated in their asocial versions. The peer is not present in the environment, and the environment is initialized so that the task can be solved alone. For example, in MARBLEPASS the marble is already on the right side of the room, so the agent just has to push it towards the *marble generator*.

Adversarial environment type This environment type is used to study the ability of the agent to infer the peer's field of view. An apple will be present in the environment right away. However, the agent will get rewarded only if it eats it while

not being observed by the peer (the peer is adversarial). Therefore, the agent needs to infer the right moment to eat the apple. There is one important parameter in this environment type. It refers to the amount of obstacles present in the environment. Figure 19 shows this environment type without any obstacles (figure 19(a)) and with obstacles present (figure 19(b)).

C.4. Architecture of the RL agent

The multimodal observation space consists of a 7x7x6 tensor (vision) and the full dialogue history (language). The multimodal action space consists of 6 primitive actions (no_op, movement actions, *toggle*, and *done*), and a 4x16 templated language.

In this work we use a PPO-trained (Schulman et al., 2017) DRL architecture initially designed for the BabyAI benchmark (Chevalier-Boisvert et al., 2019). The policy design was improved in a follow-up paper (Hui et al., 2020) (more precisely, we extend their *original_endpool_res* model). See figure 13 for a visualization of the complete architecture. First, symbolic pixel grid observations are fed into two convolutional layers (LeCun et al., 1989; Krizhevsky et al., 2012) (3x3 filter, stride and padding set to 1), while dialogue inputs are processed using a Gated Recurrent Unit layer (Chung et al., 2015). The resulting image and language embeddings are combined using two FiLM attention layers (Perez et al., 2017). Max pooling is performed on the resulting combined embedding before being fed into an LSTM (Hochreiter & Schmidhuber, 1997) with a $128D$ memory vector. The LSTM embedding is then used as input for the navigation action head, which is a two-layered fully-connected network with tanh activations and has an 6D output (i.e. 5 navigation actions and no_op action).

In order for our agent to be able to both move and talk, we add to this architecture a talking action head, which is composed of three subheads. All of them consist of two fully-connected layers with tanh activations, and take the LSTM's embedding as input. The first one is used as a switch: it has a one-dimensional output to choose whether the agent talks (output >0.5) or not (output <0.5). If the agent talks, the two other networks are used to sample the template and the word. Grammar of the templated language is depicted in table 2 and examples of multi-modal actions in table 3.

C.5. Exploration bonuses

The exploration bonuses we use are inspired by recent works in intrinsically motivated exploration (Pathak et al., 2017; Savinov et al., 2018; Tang et al., 2017). These intrinsic rewards estimate the novelty of the currently observed state and add the novelty based bonus to the extrinsic reward.

In this work we present two techniques for computing the count-based exploration bonus. Both of our count-based exploration bonuses are episodic - they estimate the diversity of states observed within an episode, and assume that beneficial episodes are those with more diverse observations. We compare those two exploration bonuses to RIDE (Raileanu & Rocktäschel, 2020), RND (Burda et al., 2018), and to the agent without any exploration bonus. [1]

Language-based exploration bonus (CBL) For some utterance s_{lang} observed at state s, we count how many times was this utterance observed during the episode. We compute the bonus for this step using the following equation:

$$r_{intr} = T * tanh\left(\frac{C}{(N(s_{lang})+1)^M}\right) \tag{1}$$

, where M, C, and T are hyperparameters and $N(s_{lang})$ is the number of times the utterance s_{lang} was observed during this episode so far.

Vision-based intrinsic reward (CB) We reward the agent for observing diverse encodings. An encoding is the 6D representation of a cell (see figure 8 for more details). A visual observation consists of 47 (7x7) encodings representing cells in front of the agent. For some visual observation s_{viz} at step s, a set of encountered unique encodings is created (duplicates are removed) $U(s_{viz})$, and then the reward computed using the following equation:

$$r_{intr} = T * tanh\left(\sum_{e \in U(s_{viz})} \frac{C}{(N(e)+1)^M}\right) \tag{2}$$

[1] We verify our implementation of RIDE and RND by recreating the results of those baselines on environments from RIDE (Raileanu & Rocktäschel, 2020).

, where M, C, and T are hyperparameters, $U(s)$ is a set of unique encodings visible in state s, and $N(e)$ is the number of times an encoding e was encountered in the current episode.

D. Detail on the baselines and exploration bonuses used

In all of our case studies, except the study with language models (sec. 5.4), we use a PPO (Schulman et al., 2017) reinforcement learning agent (as described in section C.4) with different exploration bonuses.

In a set of pilot experiments (appendix E), we compared various PPO agents (Schulman et al., 2017) with different exploration bonuses (for details see section C.5), and outlined the most promising ones to use in the following case studies. Visual count-based exploration bonus ("PPO-CB") performed best on the tasks in which language is not used, and its linguistic variant "PPO-CBL" performed best in environments with the peer giving linguistic cues. In our experiments, these exploration bonuses outperformed RND (Burda et al., 2018) and RIDE (Raileanu & Rocktäschel, 2020). Both of those two exploration bonuses are episodic. They estimate the diversity of observations in an episode and give reward proportional to that diversity. The linguistic exploration bonus uses the number of different words, and the vision-based exploration bonus the number of different encodings observed (refer to C.5 for further details).

In the case studies in sections 5.1 and 5.2 we use the "PPO-CB" exploration bonus. The case study in section 5.3 requires raw PPO for the purposes of the study, and one in section 5.4 uses LLMs as agents. In additional experiments (section G), we use PPO-CB and PPO-CBL (in those case-studies in which the peer provides linguistic feedback).

E. Pilot experiments

In our pilot studies, we encoded the peer in a way which used the mix of egocentric and allocentric vision - the peer's gaze and pointing direction were encoded in terms of absolute direction ("NSEW"). We decided to change this to fully egocentric as we found it more natural with regards to the question of socio-cognitive artificial intelligence. We believe that the best performing baselines would also perform best with purely egocentric encodings (the one we use in the rest of the paper). For that reason, and to avoid unnecessary energy spending, we do not compare with other baselines on the purely egocentric encoding.

Figure 14 compares PPO agents trained with different exploration bonuses discussed in section C.5 on two different INFORMATIONSEEKING type environments. The first environment involves the peer pointing to the correct object. Figure 14(b) shows that the best performing agent is the one levering the visual count-based exploration bonus (PPO_CB). The second environment involves the peer uttering the color of the correct object. Figure 14(a) shows that the best performing agent is the one levering the linguistic count-based exploration bonus (PPO_CBL). We conclude that PPO_CBL is the most suitable baseline for environments involving linguistic cues, and PPO_CB for the other environments.

F. Details on the environment generation parameters used in the main experiments

F.1. The Pointing experiments

The parameter trees used in this experiment are depicted in figure 27. We used the INFORMATIONSEEKING environment type described in section C.3. The INTRODUCTORY_SEQUENCE is set to EYE_CONTACT, and the CUE_TYPE to POINTING - the peer will point to the correct object after eye contact. The agent is trained on the following five problems: BOXES, SWITCHES, LEVERS, MARBLE, GENERATORS, and on the asocial version of the DOORS problem (a version without the distractor or peer). Training on this asocial version is important as it enables the agent to learn how to use a door, which is needed to evaluate generalization.

We use the PPO agent with visual count-based exploration bonus (PPO-CB).

F.2. Role reversal imitation experiments

The parameter trees used in this experiment are depicted in figure 28. We used the COLLABORATION type environments described in section C.3. We evaluate agents on role A of the MARBLEPASS task - the agent has to push the marble to the right side of the environment, from where the peer can push it to the *marble generator*.

We train the agent with the visual count-based exploration bonus (PPO_CB)

F.3. Scaffolding experiments

The parameter trees used in this experiment are depicted in figure 29. In this experiment, we use the INFORMATION SEEKING environment type with the LANGUAGE FEEDBACK cue type. We train agents on all six problems, using different values of the INTRODUCTORY_SEQUENCE and HELP parameters. We evaluate the agents on all six problems, with the most complex introductory sequence - ASK_EYE_CONTACT.

The agent denoted by "scaf_4" is trained on four different values of the INTRODUCTORY SEQUENCE parameter, and with the HELP parameter set to N (the peer will provide cues). This agent will be trained on a total of 18 different environments: six problems, and four introductory sequences. The second agent (denoted by "scaf_8") is also trained on all values of the INTRODUCTORY_SEQUENCE parameter, but it is in addition trained on both values of the HELP parameter (N and Y) - a total of 36 environments. In half of those environments (with HELP set to Y) the peer will provide the apple to the agent after the introduction (e.g. it will go to the correct box, and open it). In the other half (with HELP set to N), the peer will only provide linguistic feedback cues.

In this experiment, we use the PPO agent without an exploration bonus.

G. Additional case studies

G.1. Inferring the meaning of linguistic cues

In this section, we study the ability of the agent to infer the meaning of simple words. We follow the same procedure as in section 5.1. This case study is motivated by the experiments from cognitive science discussed in section 3.1.2. In (Carpenter et al., 1998b) infants' word understanding steadily increased in the period between 9 and 15 months after birth. We study the following questions:

- Can an RL agent learn to interpret simple utterances?
- Can the agent generalize to new situations, and infer the meaning of those utterances for objects in a new context?

The best performing agent on the linguistic environments in the pilot study was the one using the linguistic count-based exploration bonus (PPO-CBL) (see appendix E). We use this agent to address both questions.

Environments The environments are the same as those in section 5.1: the INFORMATION_SEEKING environment type, with the INTRODUCTORY_SEQUENCE set to EYE_CONTACT. The only difference is that the peer will give linguistic cues instead of pointing. We run two experiments with two different types of linguistic cues: *Color* and *Feedback*. In *Color* the peer will utter name color of the correct object. In *Feedback* the peer will utter a description of how close the agent is to the correct object: "Cold", "Medium", "Warm", and "Hot" meaning, respectively, "far", "medium", "close" and "right next to". The experimental procedure is the same as the one in section 5.1. The agent is trained on the same five problems and the asocial version of the DOORS problem.

Can RL agents learn to interpret simple utterances?

Figures 15(a) and 15(b) show the performance of the agent with the linguistic count-based exploration bonus (denoted PPO_CBL_train). We can see that the agent (PPO-CBL) solves these environments efficiently, reaching a final performance of 95.9% and 71.4% for COLOR and FEEDBACK cue types, respectively. We further analyse the performance of each separate seed for the agent trained on the FEEDBACK cue type. This is shown in figure 16 where it is visible that the agent is normally able to achieve high performance, but that there are two seeds which, due to their instability, reach a success rate of 0. This experiment shows that the agent is capable of learning to infer the meaning of simple utterances in familiar contexts.

Can the agent generalize to new situations? A more interesting question is whether that agent can infer the meaning of the same word based on a new context. Therefore, we evaluate the agent's generalization abilities in a new scenario - the DOOR problem - following the same procedure as in section 5.1.

This kind of generalization is particularly interesting as communication depends on our ability to ground words in *new* social contexts: inferring meaning by combining the convention associated to a word with the recursively inferred intention of the speaker. For example, while "red" can mean "open the red box" in one context, it can mean "push the marble towards the red generator" in another.

Figures 15(b) and 15(a) show the performance of the same agent evaluated on the DOORS problem (denoted "PPO_CBL_test") They show that neither of the agents is capable of such generalization, which is consistent with the experiments with the pointing gesture in section 5.1.

These results motivate future research on what kind of biases could be built into the agents (and in what way) so that they could infer the meaning of familiar words in new contexts. For example, an interesting avenue of future work is to try to combine an agent with a large language models, and see if the knowledge contained in it could make the agent generalize better.

G.2. Joint Attention

Tomasello describes joint attention as consisting of two parts: triangulation and recursiveness (Tomasello, 2019). He argues that joint attention plays a key role in the 9-month revolution by transforming dyadic interactions (e.g. mimicking facial expressions) to triadic (e.g. imitating an action on an object). Joint attention was also required in the previous experiments (sections 5.1 and G.1). The agent and the peer triangulated on an external referent, however, the agent could assume that the peer was participating in the interaction.

In this experiment, we aim to conduct a more thorough test of the second aspect of joint attention - *recursiveness* (both participants being aware that they are both sharing attention). To solve the task, the agent needs to infer if the peer is participating and is aware that the agent is participating too. We create environments where the peer, in addition to giving regular cues inside joint attention, gives *misleading* cues outside joint attention. These cues are implemented uttering a random cue, and are given before the agent completes the introductory sequence. In other words, the agent should learn to discriminate between cues given for the agent during joint attention (after the introduction) and cues given regardless of the agent outside joint attention (before the introduction).

We study the following question:

- Can RL agents learn to differentiate between cues given inside and outside joint attention, i.e. can they learn to infer whether the peer is participating in the interaction?

Environments In this section, we extend the environment from section G.1 studying the COLOR cue type. The environment is extended so that the agent must also recursively infer whether the cue is intended for the agent. This misleading cue is given before the introductory sequence is completed, and takes the form of the peer uttering a color of a random one of the two objects. Apart from that, the experiments are conducted in the same way as in section G.1 (we train on the same problems for and use the same baseline).

Results Figure 17 compares the performance (success rate) of the agent trained on this extended environment (denoted by JA) with the agent trained on the regular environment (from the experiment in section G.1). These results show that the agent is not able to differentiate between cues given inside and outside of joint attention. We believe that this is due to the cues being highly misleading in this environment. As the peer utters the color of a random object present in the environment, there is a 50% possibility of the misleading cue being the same as the helpful one.

These results open many avenues for future research. One might study which kinds of biases can be integrated into the agent to make such cues less misleading. The generalization abilities of those agents should also be investigated. For instance, we could study if an agent that learned to ignore misleading linguistic cues would ignore misleading pointing cues.

G.3. Imitation learning

In the following section, we study the ability of the agent to learn how to obtain the apple by imitating the peer. This experiment is motivated by an experiment from (Carpenter et al., 1998b) discussed in section 3.1.3. In it, infants showed a steady increase in imitation learning abilities in the period between 9 and 15 months after birth. We want to test the agent's ability to imitate an instrumental action on an object.

From an AI perspective, this can be seen as meta-imitation learning. We want to see if an agent can obtain (through gradients) the imitation learning mechanism, which it could then use (during the episode) to learn how to use a new object. In this section, we study the following questions:

- Can RL agents learn (through gradients) an imitation mechanism?

In these experiment, we use the agent with the visual count-based exploration bonus (CB), as we found it worked best in our pilot study (see appendix E). We compare three agents trained with the same exploration bonus scaled by different weights: 0.25, 0.5, and 1.

Environment The Environment is an INFORMATION SEEKING type environment without a distractor. After the introductory sequence (EYE_CONTACT), the peer will demonstrate using an object to obtain the apple. For example, it will toggle a box or push a generator. Then the peer will then eat the apple, and revert the environment to its initial state. The agent should then imitate the peer - use the same action on the object - to obtain the apple for itself. If the agent uses the object it in the wrong way (e.g. pushes the box instead of toggling it) it will be blocked, and the apple will not be obtainable in this episode. The agents are evaluated on a new problem in which the agent encounters a new object for the first time. This means that the agent must pay attention to how the peer uses the object, and use it in the same way. [2]

The agents are trained on five problems (all expect DOORS). Most importantly, compared to the experiments in sections 5.1 and G.1, these agents will not be trained on the asocial version of the DOORS problem. That is because, in the generalization testing, we want to see if the agent can learn to use a completely new object.

Results Figure 18 shows the performance (success rate) of the agents on the training environments, the percentage of succesful introduction with the peer, and the evaluation on the (unseen) DOORS problem.

On figures 18(a) and 18(b) we can see that the agent with a lot of exploration bonus (PPO_CB_1) is too focused on the peer and, and is unable to solve the task. This is implied by the high percentage of the successful introductory sequence, and low success rate on the training environments. On the other hand, the agent with smaller exploration bonus weight (PPO_CB_0.25) solves these environments without problems, however it does use the peer. As such, the agent can solve the training environments by ignoring the peer and discovering how to use each object by itself. However, this agent is not able to generalize to a new object as the only way to know how to use that object is to observe the peer's demonstration (see figure 18(c)). Figure 18(c) shows the performance of those agents on the testing environment. The figure shows that neither of the three agents is capable of acquiring a meta-imitation learning mechanism that can generalize to a novel object (marble).

These results are not surprising, as current exploration bonuses are not well suited to enable RL agents to meta-learn mechanisms. These results imply that an interesting avenue of research is to study how to endow agents with such meta-imitation learning mechanisms that would enable them to learn a behavior in a new scenario. An promising solution to this problem are large language models and other large transformer-based networks pretrained on many other tasks. It would be interesting to study if such agents already have an imitation learning mechanism which would enable such online imitation. This would open up countless avenues of research into various forms of online imitation and emulation learning.

G.4. Inferring another's field of view

In this section, we study the ability of the agent to infer what the other observes. This experiment is modeling the one in Hare (2001). In it, apes were shown to be able to infer what another sees, as they only took the food the alpha male could not see.

In this section, we want to study the following questions:

- Can agents learn to infer the other's field of view?

Environment In the following experiment, we are using the ADVERSARIALPEER environment type, in which the agent has to eat the apple while not being seen by the peer. We study two versions of this environment: with and without obstacles (for more details, refer to section **??**). Obstacles make the problem of inferring the peer's field of view harder.

Experiment We study how the agent infers the peer's field of view by training the agent on the AdversarialPeer task.

It is important to note that this agent can sometimes use other (asocial) information to achieve performance. For example, if the object is surrounded by occlusions the agent could guess that it is not observed by the peer, which is not necessarily the case. To better understand the performance of the agent we compare the agent with two baselines. First, we assess to what extent the agent is making inferences based on the peer's location and gaze direction. We train an agent ("invisible_peer") that has the peer filtered from its observations (it cannot observe the peer). This baseline estimates the maximum possible

[2]The encoding of the peer includes the peer's previous timestep action.

performance If the standard agent outperforms this baseline this implies that it is leveraging the social information in the environment. Second, to estimate the upper bound on the performance we train an agent in the environment without the peer present (this agent is reward every time it eats the apple).

Results Figure 20 shows the performances of those three agents. It shows that the agent outperforms the agent with the peer filtered from its observations ("invisible_peer"), which implies that the agent is using the peer's location and gaze direction to infer whether to eat the apple or not. Furthermore, the agent is not able to match performance of the agent trained without the peer present in the environment ("no_peer"). This results imply that, while the agent is able to leverage some social information in the environment, there still remains room for improvement. Future research could focus on constructing novel types of exploration bonuses to bridge this gap.

G.5. Formats

In the following experiment, we study the ability of the agent to learn formats (also referred to as pragmatic frames in (Vollmer et al., 2016)). Formats are a concept introduced by Jerome Bruner, which we discussed in more detail in section 3.2. They can be regarded as protocols of social interactions. We study the following question:

- To what extent can an exploration bonus help with the acquisition of a complex format.

We address this question by training two agents (one with an exploration bonus, and one without it).

Environment We use the INFORMATION SEEKING environment type with the LANGUAGE FEEDBACK cue type. We train all agents on all six problems. In contrast to section G.1, where the introductory sequence was always set to EYE CONTACT, here it is set to ASK EYE CONTACT - the peer will give cues after the agent utters "Help, please" during eye contact.

Results Figure 21 compares the performance of an agent that does not use any exploration bonus ("PPO_no_bonus") to an agent that uses the visual count-based exploration bonus ("PPO_CBL"). The agent with the exploration bonus achieves high performance (97.9% success rate) and greatly outperforms the agent without the exploration bonus. These experiments show that, as expected, learning complex formats can be made easier with exploration bonuses.

This experiment can be interpreted in tandem with the experiment in section 5.3 where we show how more complex formats can be learned by weaker agents (without an exploration bonus) when learning in a scaffolded environment. Future work could explore how these two different approaches - modifying the agent and modifying the environment - can be used in tandem to learn even more complex formats. Furthermore, one interesting research direction is to study which kinds of problems are better addressed by modifiying the agent and which by modifying the environment.

G.6. Additional information on the case study with large language models as interactive agents

Figures 25 and 26 show the in context examples given to large language models in section 5.4. The in context examples were created by hand.

H. Extended discussion

In this work, we outline and discuss several concepts from developmental psychology – mostly regarding the development before and around 9 months of age – which we found to be most relevant for AI at the moment. Even among this restricted set it is not reasonable to aim for an exhaustive introduction. As such, several socio-cognitive concepts are either discussed very briefly (e.g. conformity, social norms, instructed learning) and a lot of others are not mentioned (e.g. morality, fairness, sense of self). We leave their analysis for future work. Furthermore, while we argue that the work of Tomasello and Bruner provides an interesting framework to guide AI research in social skill acquisition, many other perspectives could have been considered, e.g. Erik Erikson (Erikson, 1993), Alison Gopnik (Gopnik & Meltzoff, 1997), or Cecilia Heyes (Heyes, 2019).

Similarly, as the present work merely represents a first step towards socially proficient artificial learners, many technical dimensions were simplified. In particular, we refrain from free form language dialogues and consider simple templated language. Likewise, we do not use human or trained peers, but scripted peers (which enables to isolate social abilities). Rather than implementing rich 3D visual worlds with continuous actions, we use grid-worlds with discrete primitive actions. We argue that such simplifying assumptions affords tractable studies while maintaining enough social complexity to model and isolate various social challenges. Assuming progress is made over these social scenarios, an interesting avenue for

Table 2. Template-based grammar used in all of the SocialAI environments. If the agent decided to speak it chooses a template and a noun to insert into the template.

Nouns		
Action	Template	Noun
0	Where is <noun>	please
1	Help <noun>	the exit
2	Close <noun>	the wall
3	How are <noun>	you
4		the ceiling
5		the window
6		the entrance
7		the closet
8		the drawer
9		the fridge
10		the floor
11		the lamp
12		the trash can
13		the chair
14		the bed
15		the sofa

Table 3. Examples of actions in the environment. Second and third dimension must both either be underfined or not. In practice, there is an additional binary output which defines if the agent will speak.

Action	description
(1, -, -)	moves left without speaking
(1, 1, 5)	moves left and utters ”Help the window”
(-, 1, 5)	doesn’t move but utters ”Help the window”
(-, -, -)	nothing happens

future work will be to extend the parametric generation towards environments with more complex sensorimotor challenges.

Given recent works showcasing the importance of Automatic Curriculum Learning in "asocial" DRL (Parker-Holder et al., 2022; Portelas et al., 2020), an interesting direction for future work would be to study whether this can also be observed in SocialAI. Our short case study on the importance of scaffolding (sec. 5.3) suggests a positive impact, although we restricted our analysis to simple expert curricula. An important challenge will be to design curriculum methods able to leverage the hierarchical structure of SocialAI's parametric tree, rather than the usual low-dimensional flat spaces of task-encoding parameters (predominant in the literature).

Large language models (LLMs) are present in many branches of artificial intelligence. A promising avenue of future research is the application of language models to interactive agents (Andreas, 2022). In this paper, we studied LLMs only on simple environments with a simple method - prompting the model with a few expert trajectories. While this approach showed impressive sample efficiency, it is very limited due to the constraints on the prompt size. These experiments should be revisited with more powerful methods such as fine-tuning or chain-of-thought prompting. Such methods could potentially make more complex social inferences, leading to better performance on many case studies in this paper, especially the ones related to generalization to new scenarios.

An important factor for the observed learning failures of our PPO agents in our case-studies might be linked to the simple forms of exploration bonuses that we used (variants of count-based bonuses (Bellemare et al., 2016)). Finding efficient exploration bonuses for social settings is a challenging task. In appendix E we show that RIDE and RND (Raileanu & Rocktäschel, 2020; Burda et al., 2018), two state-of-the-art exploration bonuses from classical DRL underperformed compared to our simple methods. An interesting avenue would be to study the robustness of recent exploration bonus methods designed for social scenarios (e.g. Zhang (2020)).

A limitation of this work is that, in our experiments with reinforcement learning, we only consider agents learning through pure extrinsic or intrinsic reward. An interesting avenue for future work is to study how existing imitation learning methods such as GATO (Reed et al., 2022) or the decision transformer (Chen et al., 2021) could acquire social skills better if provided with expert demonstrations and trained jointly on other tasks which require social competence (e.g. Sap (2019), Netanyahu (2021), Shu (2021)). Another promising line of works are methods combining reinforcement learning signals with imitation learning (Hester et al., 2018; Nair et al., 2018).

Another important limitation of the present work is the lack of human-AI interactions during learning. In light of human social learning, such interactivity appears like an important step towards socially proficient learners. While we argue that scripted peers constitute a good basis for initial progress, it is crucial for long-term progress that more interactive learning paradigms are considered inside SocialAI. (Abramson et al., 2021) constitutes a noteworthy step towards this goal. On a more general note, featuring human inputs inside the learning pipeline of autonomous learners could constitute an efficient way to instill human-like inductive biases inside agents, which is an important feature to align agents' cognition to ours. For instance, in Kumar (2022) the authors show in a 2D-tile revealing task that autonomous agents learning from human-generated language description reached performances more similar to humans compared to agents trained with synthetic language.

From a high level perspective, the SocialAI school presents many relevant concepts (e.g. skills, motivation) which are at the core of human cognition. We discussed in more details those we considered most relevant for AI at the moment. We do not claim to have covered every relevant aspect of early social cognition: many other dimensions remain to be studied, both from the work of Tomasello and Bruner and from other authors. The SocialAI school is a tool which can be easily extended and modified to study many of these aspects. We hope this paper will motivate the AI community around the questions of social cognition. This avenue of research, on a longer timeline, promises to endow AI with the abilities and motivations enabling it to enter and push our culture forward, i.e. participate in our cumulative cultural evolution.

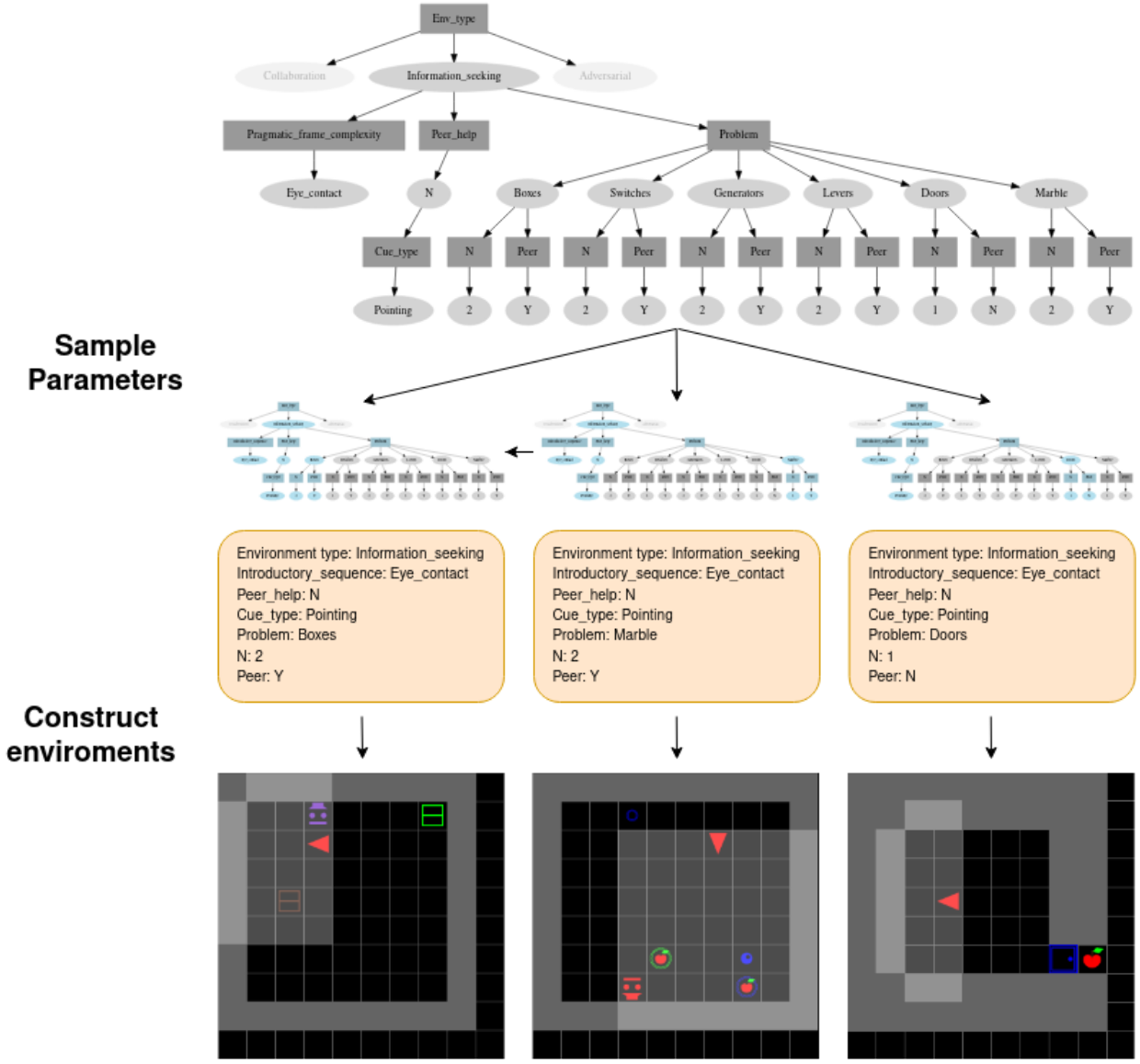
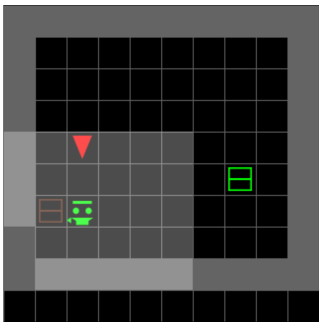
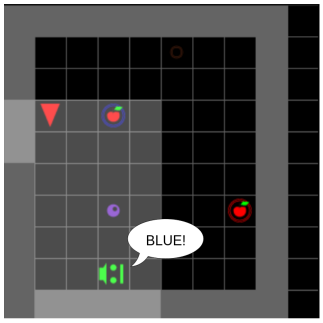


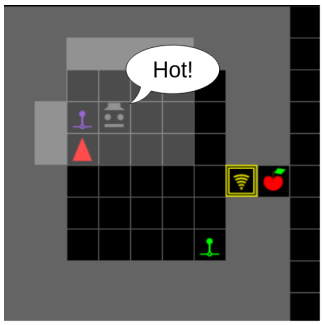
Figure 10. An example of procedural environment generation using tree-based parametric sampling. There are two kinds of nodes: parameter nodes (rectangles) and value nodes (ovals). Parameter nodes require that one of its children (a value node) is selected. Value nodes require that sampling progresses through all of its children (parameter nodes). In this tree, all parameter nodes except "Problem" have only one child. This means that only the Problem parameter can be set in different ways. We show three examples of parameter sampling, and the three environments constructed from those parameters.



(a) A scripted peer pointing to a box. The agent needs to open the red box.

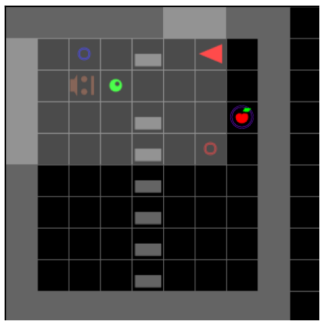


(b) A scripted peer uttering the color of the correct generator. The agent needs to push the marble onto the blue generator.

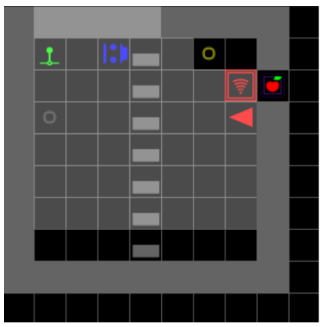


(c) A scripted peer hinting the distance to the correct lever ("Hot" means very close). The agent needs to pull the purple lever to open the door.

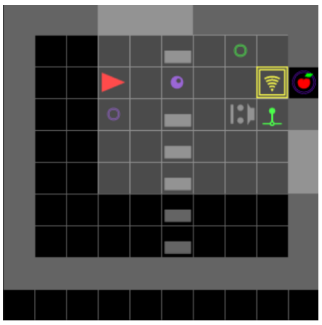
Figure 11. Examples of INFORMATIONSEEKING type environments, in which agents learns to find hidden apples using textual or non-verbal communication with social peers.



(a) The MARBLEPASS problem with the agent in role B. The peer pushes the marble to the right and then the agent pushes it further to the purple *marble generator*. This makes two apples appear on the blue and red platforms.



(b) The LEVERDOOR problem with the agent in role B. The peer opens the red door by pulling on the green lever. This enables the agent to go through the door and activate the purple generator This makes two apples appear on the gray and yellow platforms.



(c) The MARBLEPUSH problem with the agent in role A. The peer opens the yellow door using the green lever. Then the agent pushes the marble through the door to the purple *marble generator*. This makes two apples appear on the purple and green platforms.

Figure 12. Examples of COLLABORATION type environments, in which agents must learn cooperative strategies with a (scripted) peer to achieve two-player puzzles.

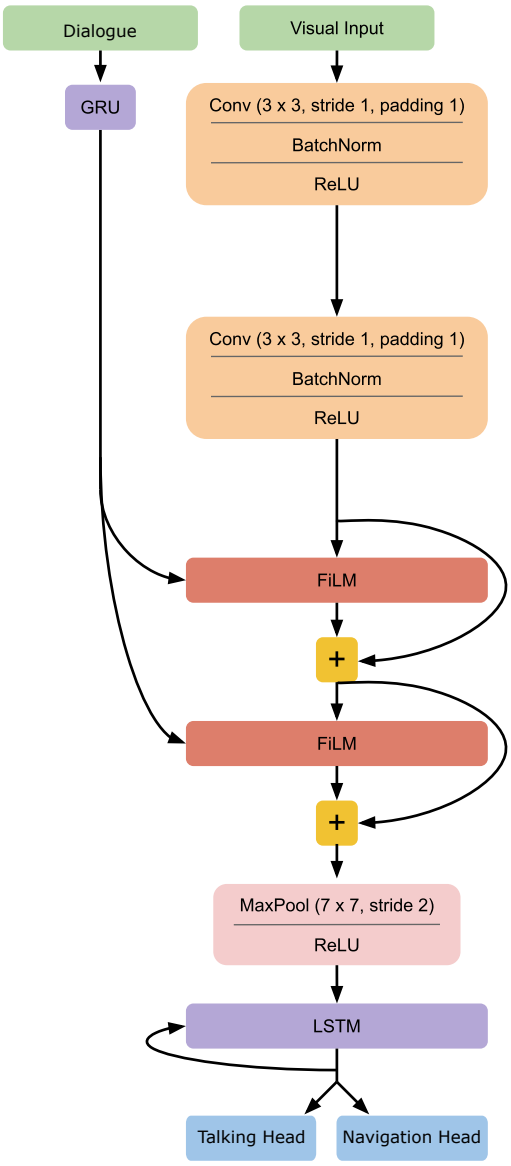
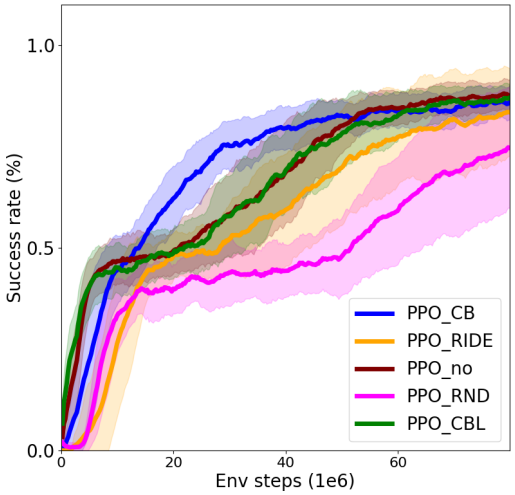
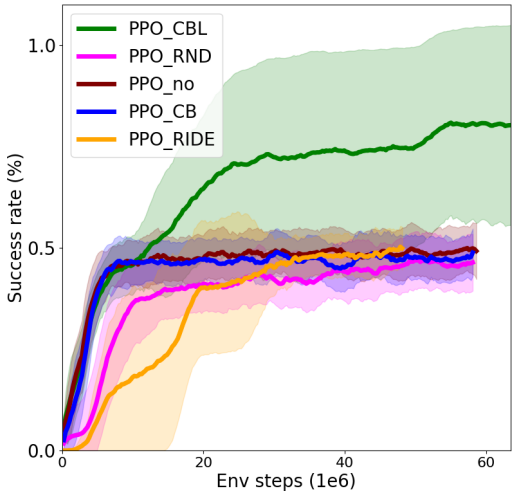


Figure 13. Our Multi-Headed PPO baseline DRL agent. Architecture visualization is a modified version of the one made by Hui (2020). We perform two modifications: 1) Instead of fixed instruction inputs our model is fed with NPC's language outputs (if the agent is near an NPC), and 2) We add a language action head, as our agent can both navigate and talk.

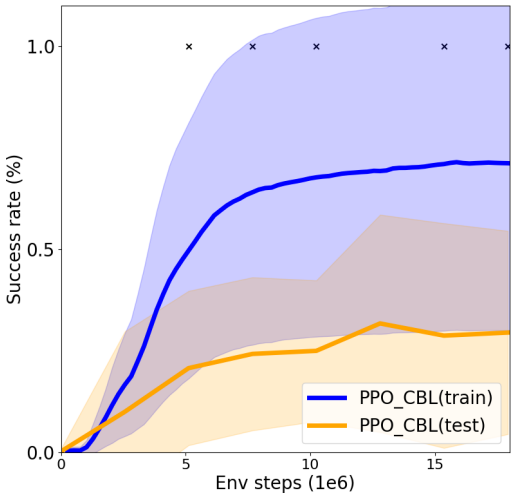


(a) Pilot experiments with the peer pointing to the correct object.

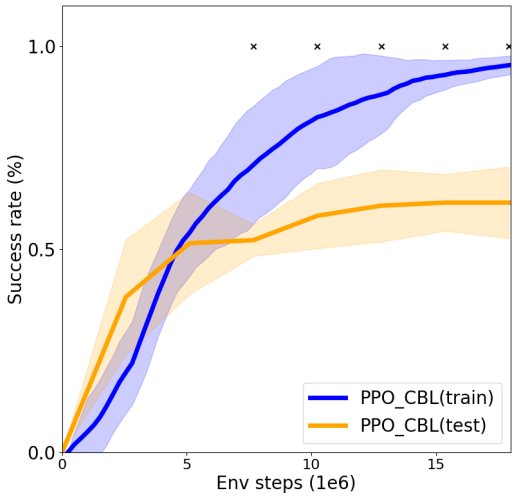


(b) Pilot experiments with the peer uttering the color of the correct object.

Figure 14. Pilot experiments showing that our count-based exploration bonuses outperform other baselines. On the environments with the pointing gesture, visual count-based ("CB") exploration bonus is the best performing condition. On the environments with utterances, linguistic count-based ("CBL") exploration bonus is the best performing condition.



(a) Language Feedback cue type experiments: the peer gives cues regarding the proximity of the agent is to the correct object (e.g. Hot, Warm, Cold).



(b) Language Color cue type experiments: the peer utters the color of the correct object.

Figure 15. **The linguistic cues experiments.** We study if an RL agent is able to infer the meaning of linguistic cues in order to use the correct object. We consider two types of cues: *language feedback* and *color*. In both settings, the agent was trained on five different problems, and on the asocial version of the Doors problem (only one door and no peer present in the environment) - denoted by "train". Agents were periodically evaluated on the social version of the Doors problem (two doors and a peer giving cues) - denoted by "test". The figure compares the success rate (mean +/- std over 8 seeds) on the training environments with the evaluation on the testing environment. The cross marks depict statistical significance ($p = 0.05$). In both cases the agents achieve much better performance on the training problems, but fail to generalize to a new problem - the agent is not able to infer the meaning of an utterance in a new context.

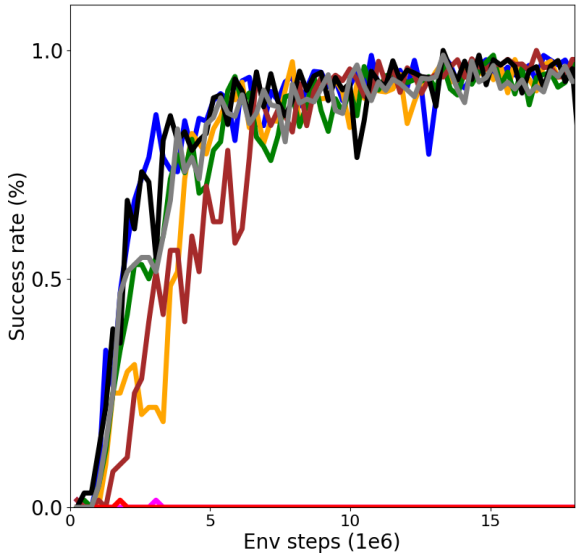


Figure 16. Per-seed performance on the training environments of the agent from figure 15(a) (”PPO_CBL(train)”). The figure shows that the agent is able to solve the training tasks efficiently, but that there are two unstable seeds which result in the success rate of 0%.

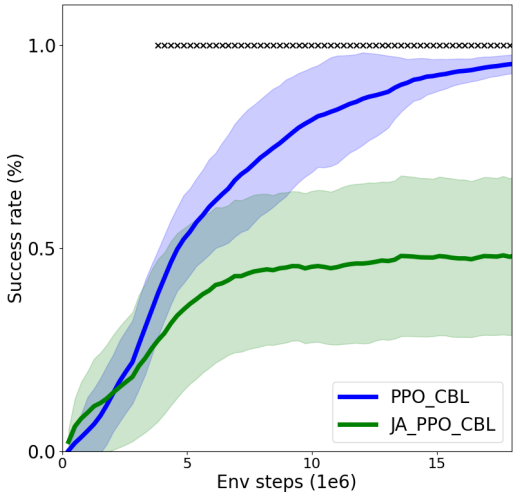
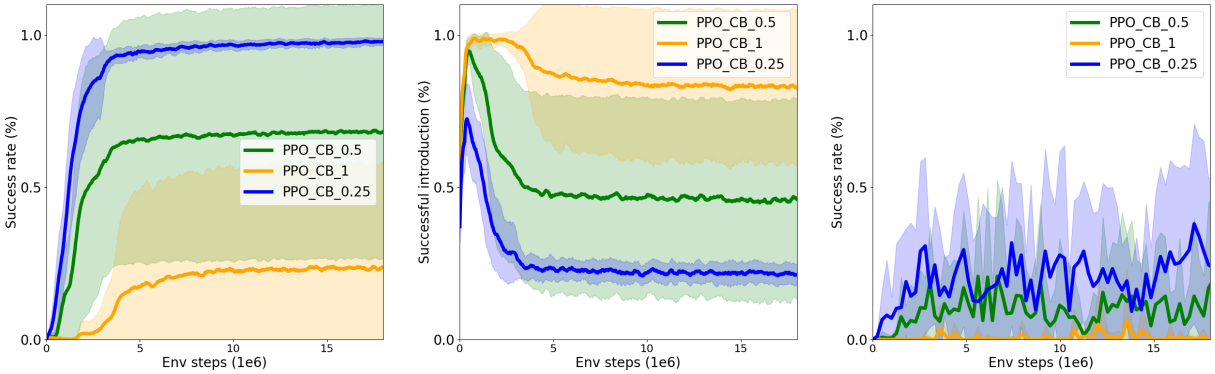


Figure 17. **The joint attention experiment**. The environments feature a test for recursiveness - infer if the peer knows that they are working together. The environments are same as the ones from figure 15(b), but with the addition of misleading cues - random cues given regardless of the agent (a random color). The peer gives misleading cues outside of joint attention (before the introductory sequence). The agent should ignore these cues, and use only cues given inside joint attention. The figure compares the success rate (mean +- std over 8 seeds) of the agent trained on the environments with both regular and misleading cues(”JA_PPO_CBL”), to the agent trained on the environments with only regular cues (”PPO_CBL(train)” from figure 15(b)). The figure shows that the agent unable to master the Joint Attention variant.

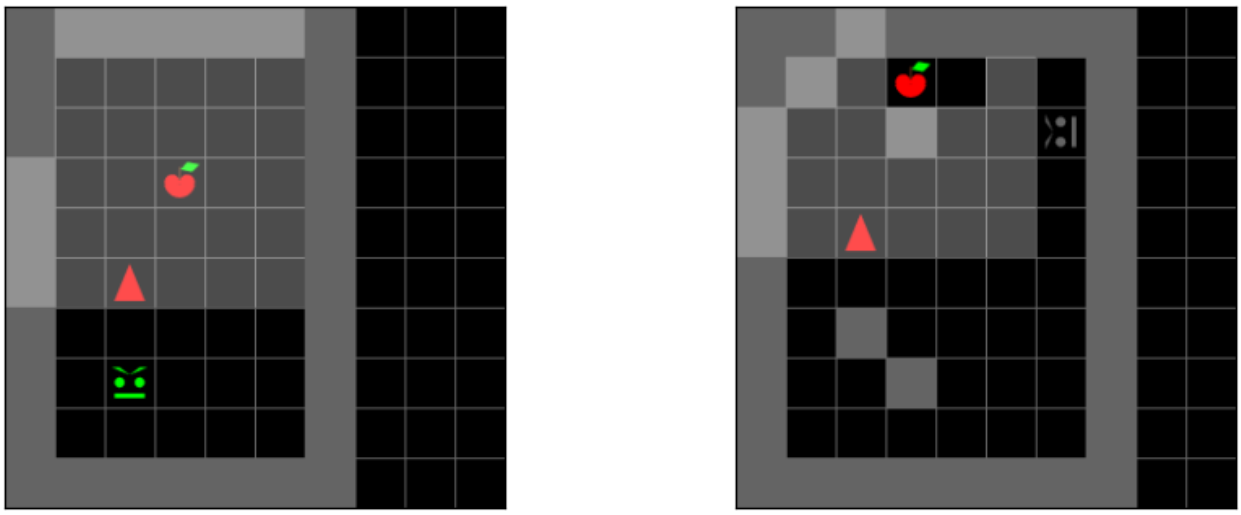


(a) Imitation experiments performance (success rate) on the training environments.

(b) The percentage of successful introductory sequences on the training environments.

(c) Imitation experiments performance (success rate) on the testing environment.

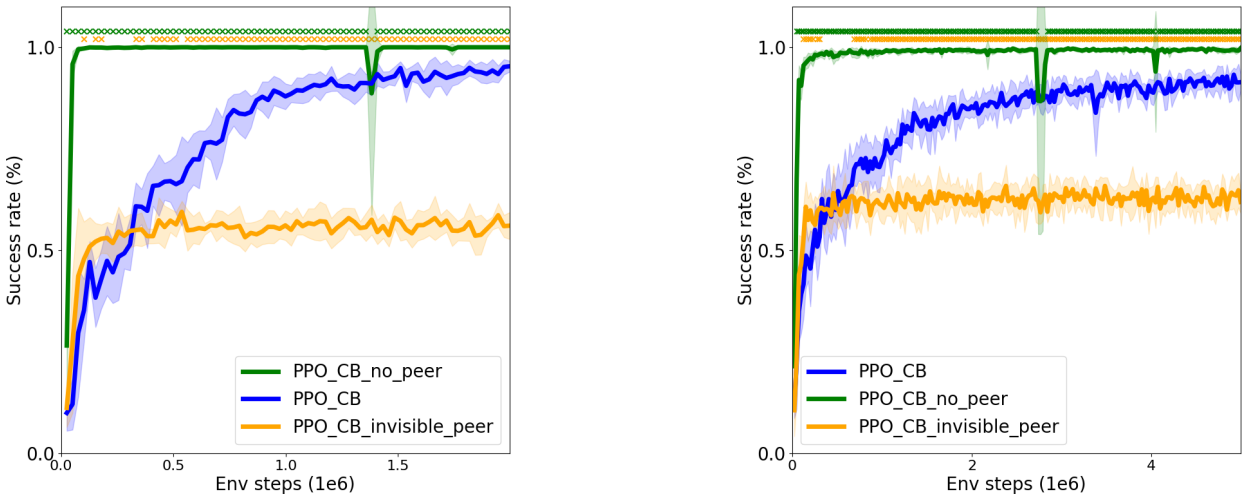
Figure 18. **Imitation learning experiments.** The peer demonstrates how to use an object (after the agent successfully introduces itself). The agent is trained on five different problems and evaluated on a new problem with a previously unobserved object (a door). A socially proficient agent should be able to learn (by observing the demonstration) which action (toggle or push) to use on the new object. The curves compare three agents trained with a different scaling factor for the visual count-based exploration bonus. One can see that the agent with high exploration bonus ("PPO_CB_1") focuses too much on the peer, which results in ignoring the task. This is evidenced by high success in completing the introductory sequence (fig. 18(b)), but low success rate on the task (fig. 18(a)). On the other hand, using low exploration bonus ("PPO_CB_0.25") pushes the agent to solve the training task whilst ignoring the peer. Rather than observing the peer's demonstration, this agent learns how to use objects by themselves. This results in perfect performance on the training object, but it makes it impossible to generalize to a new object. Neither of the agents is able to achieve high performance on the held out testing environment. This implies that they are not able to learn (online) through imitation which action to use with a new object.



(a) Adversarial peer environment without occlusions.

(b) Adversarial peer environment with occlusions.

Figure 19. Environments from the Adversarial peer experiments in which the agent has to infer the peer's field of view. The agent is rewarded upon eating the apple on the condition that it was not in the field of view of the peer while doing so. We run the experiments with two different settings: with and without occlusions (depicted in figures 19(b) and 19(a)). Occlusions make it harder to infer the peer's field of view as it is no longer rectangular.



(a) No occlusions. (b) Occlusions

Figure 20. Adversarial peer experiments. We compare three agents on two environments (depicted on figures 19(b) and 19(a)). The "PPO_CB" agent is trained on the regular environment (rewarded upon eating the apple while not being observed by the peer). The "PPO_CB_no_peer" agent is trained in the environment without the peer (the agent is rewarded every time it eats the apple). This represents the upper bound of the performance. The "PPO_CB_invisible_peer" agent is trained on the regular environment with the peer filtered from the agent's observations. This represents the performance of a completely asocial agent which ignores the peer. Figures 20(a) and 20(b) compare the performance of these three agents (8 seeds +- std), the crosses depict a statistically significant difference (p¡0.05) compared to the "PPO_CB" agent. The results show that the "PPO_CB" agent is able to partially infer the peer's field of view (as it outperforms the "invisible_peer" baseline), but is not able to reach perfect performance (as defined by the "PPO_CB_no_peer" baseline).

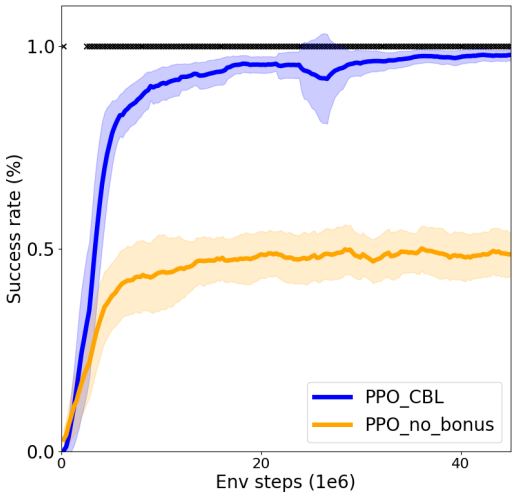


Figure 21. Comparison of an agent with and without the exploration bonus on an environment with a more complex introductory sequence (format). The task consists of the agent doing the introductory sequence by making eye contact and uttering "Help, please". The peer will then give linguistic cues regarding the proximity of the agent to the target object (e.g. Hot, Warm, Cold). Based on these cues, the agent should use the target object, instead of the distractor, to obtain the apple. The figure shows that using the visual count-based exploration bonus enables the agent to learn a more complex introductory sequence and solve the task.

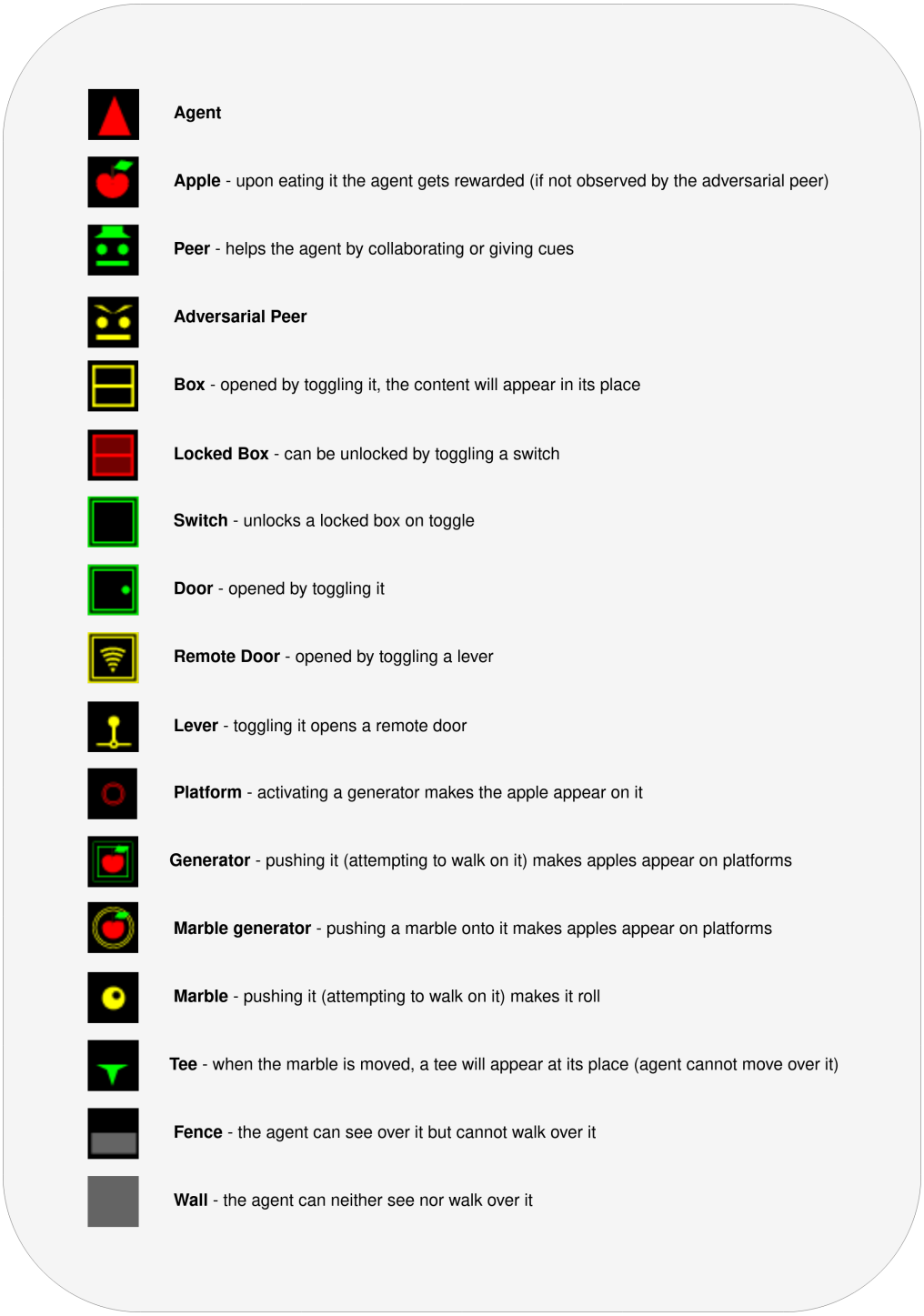
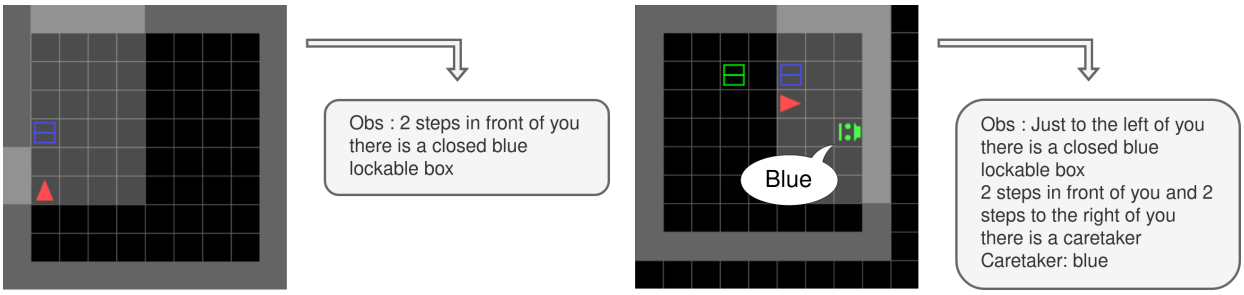


Figure 22. Visualizations and descriptions for all objects featured in SocialAI environments.



(a) The AsocialBox environment (b) The ColorBoxes environment

Figure 23. Two environemnts used in the experiments with large language models. The observations are parsed into pure text.

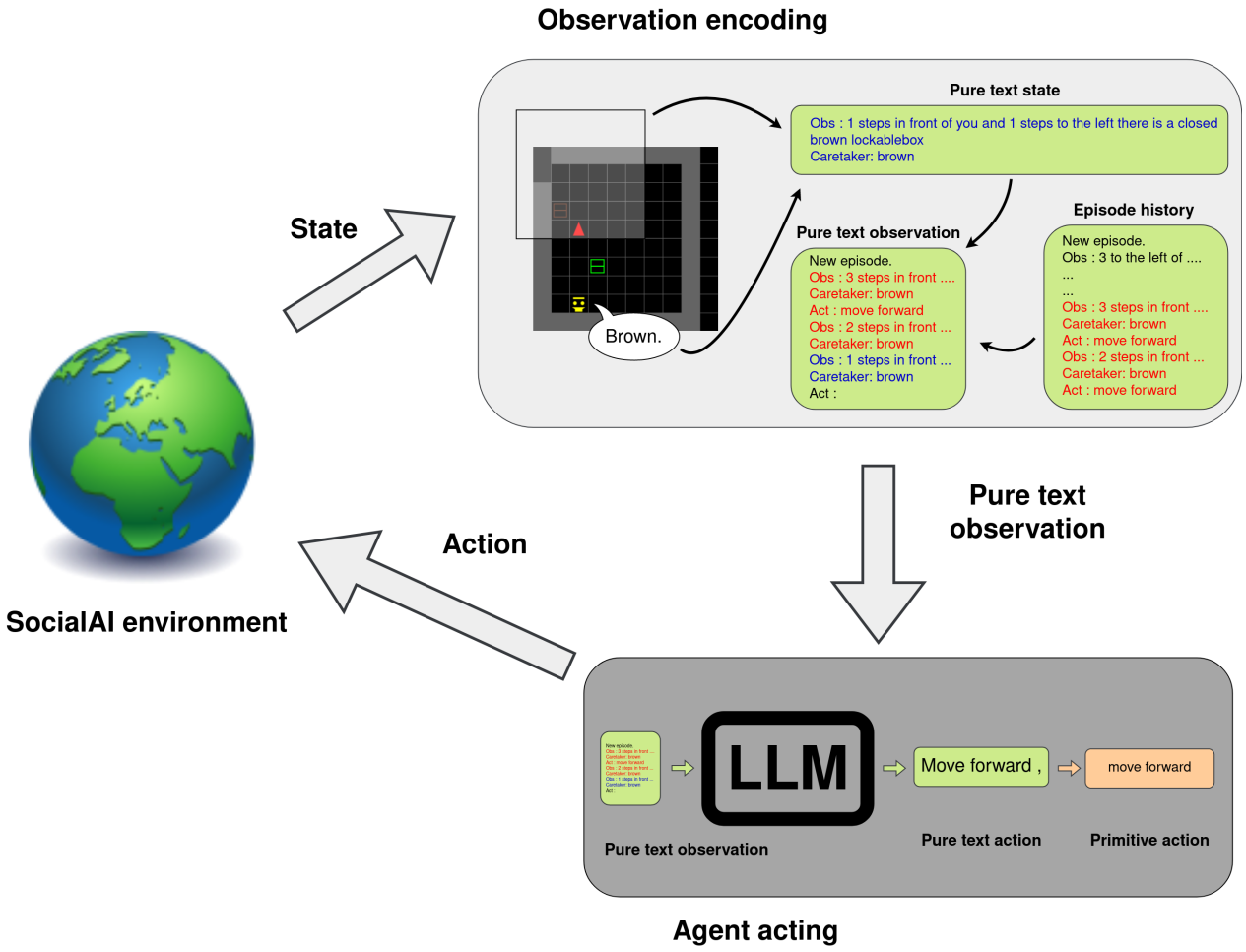


Figure 24. An example of how a language model can be used as an interactive agent in SocialAI. A state is parsed into a pure text observation and combined with previous two observations and actions. This is, appended to the in context examples, is used as prompt for the LLM. The agent generates the text which is then matched (as case insensitive substring) with the list of possible actions. The matched action is executed in the environment.

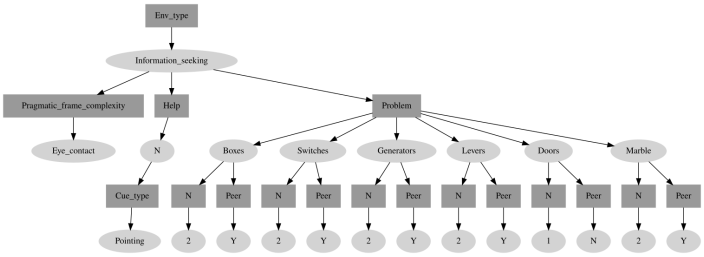
```
New episode.
Obs : 3 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : 2 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
New episode.
Obs : 2 steps to the right there is a closed green lockablebox
Act : turn right
Obs : 2 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
New episode.
Obs : 2 steps to the left there is a closed green lockablebox
Act : turn left
Obs : 2 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
New episode.
Obs : 2 steps in front of you and 2 steps to the right there is a
      closed blue lockablebox
Act : move forward
Obs : 1 steps in front of you and 2 steps to the right there is a
      closed blue lockablebox
Act : move forward
Obs : 2 steps to the right there is a closed blue lockablebox
Act : turn right
Obs : 2 steps in front of you there is a closed blue lockablebox
Act : move forward
Obs : Right in front of you there is a closed blue lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
```

```
New episode.
Obs : 2 steps in front of you and 1 steps to the left there is a
     closed green lockablebox
Act : move forward
Obs : 1 steps in front of you and 1 steps to the left there is a
     closed green lockablebox
Act : move forward
Obs : Just to the left of you there is a closed green lockablebox
Act : turn left
Obs : Right in front of you there is a closed green lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
New episode.
Obs :
Act : turn right
Obs : 1 steps in front of you and 1 steps to the right there is a
      closed blue lockablebox
Act : move forward
Obs : Just to the right of you there is a closed blue lockablebox
Act : turn right
Obs : Right in front of you there is a closed blue lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
New episode.
turn right
Obs :
Act : turn right
Obs :
Act : turn right
Obs :
Act : move forward
Obs :
Act : turn right
Obs : 4 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : 3 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : 2 steps in front of you there is a closed green lockablebox
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Act : toggle
Obs : Right in front of you there is a red apple
Act : toggle
Success!
```

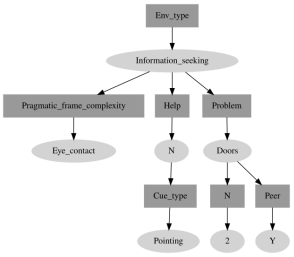
Figure 25. In context examples for the SocialAIBox environment.

```
New episode.
Obs : 3 steps in front of you there is a closed green lockablebox
Caretaker: green
Act : move forward
Obs : 2 steps in front of you there is a closed green lockablebox
Caretaker: green
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Caretaker: green
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs : 2 steps to the right there is a closed green lockablebox
Caretaker: green
Act : turn right
Obs : 2 steps in front of you there is a closed green lockablebox
Caretaker: green
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Caretaker: green
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs : 2 steps to the left there is a closed green lockablebox
Caretaker: green
Act : turn left
Obs : 2 steps in front of you there is a closed green lockablebox
Caretaker: green
Act : move forward
Obs : Right in front of you there is a closed green lockablebox
Caretaker: green
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs : 2 steps in front of you and 2 steps to the right there is a
      closed blue lockablebox
Caretaker: blue
Act : move forward
Obs : 1 steps in front of you and 2 steps to the right there is a
      closed blue lockablebox
Caretaker: blue
Act : move forward
Obs : 2 steps to the right there is a closed blue lockablebox
Caretaker: blue
Act : turn right
Obs : 2 steps in front of you there is a closed blue lockablebox
Caretaker: blue
Act : move forward
Obs : Right in front of you there is a closed blue lockablebox
Caretaker: blue
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: blue
Act : toggle
Success!
New episode.
Obs : 2 steps in front of you and 1 steps to the left there is a
     closed green lockablebox
Caretaker: green
Act : move forward
Obs : 1 steps in front of you and 1 steps to the left there is a
     closed green lockablebox
Caretaker: green
Act : move forward
Obs : Just to the left of you there is a closed green lockablebox
Caretaker: green
Act : turn left
Obs : Right in front of you there is a closed green lockablebox
Caretaker: green
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs :
Caretaker: blue
Act : turn right
Obs : 1 steps in front of you and 1 steps to the right there is a
      closed blue lockablebox
Caretaker: blue
Act : move forward
Obs : Just to the right of you there is a closed blue lockablebox
Caretaker: blue
Act : turn right
Obs : Right in front of you there is a closed blue lockablebox
Caretaker: blue
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: blue
Act : toggle
Success!
New episode.
Obs : 3 steps in front of you and 3 steps to the left there is a
     friendly yellow peer. It is looking towards you.
1 steps to the right there is a closed brown lockablebox
Caretaker: brown
Act : turn right
Obs : Right in front of you there is a closed brown lockablebox
Caretaker: brown
Act : toggle
Obs : Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs : 1 steps to the right there is a closed blue lockablebox
Right in front of you there is a closed brown lockablebox
Caretaker: brown
Act : toggle
Obs : 1 steps to the right there is a closed blue lockablebox
Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
New episode.
Obs : 1 steps to the right there is a closed blue lockablebox
Right in front of you there is a closed brown lockablebox
Caretaker: brown
Act : toggle
Obs : 1 steps to the right there is a closed blue lockablebox
Right in front of you there is a red apple
Caretaker: green
Act : toggle
Success!
```

Figure 26. In context examples for the ColorBoxes environment.

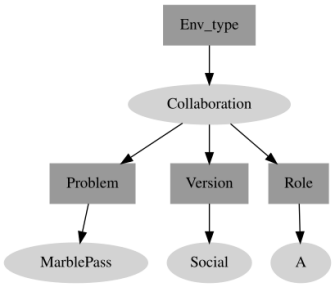


(a) Training sampling tree

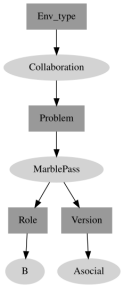


(b) Testing sampling tree - Social Doors

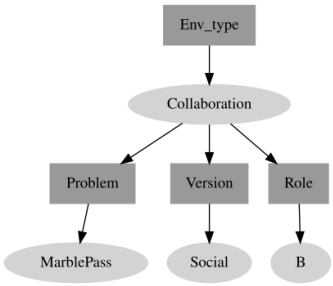
Figure 27. Sampling trees used in the pointing case study in section 5.1



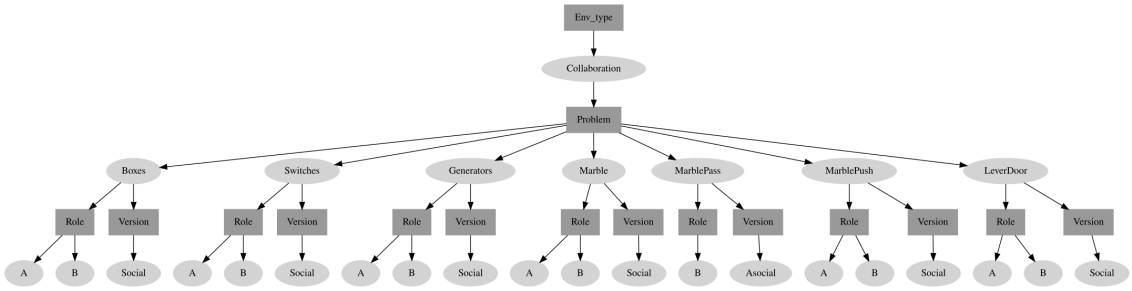
(a) Role A



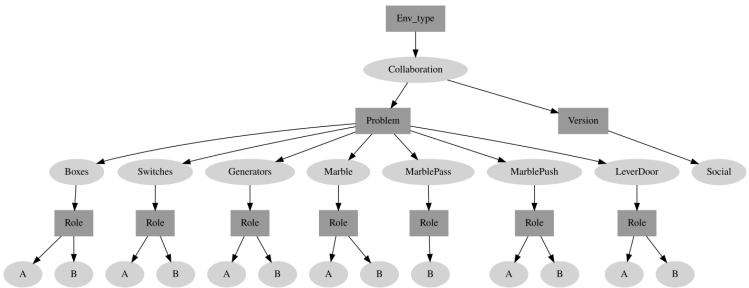
(b) Asocial single setting



(c) Role B single setting

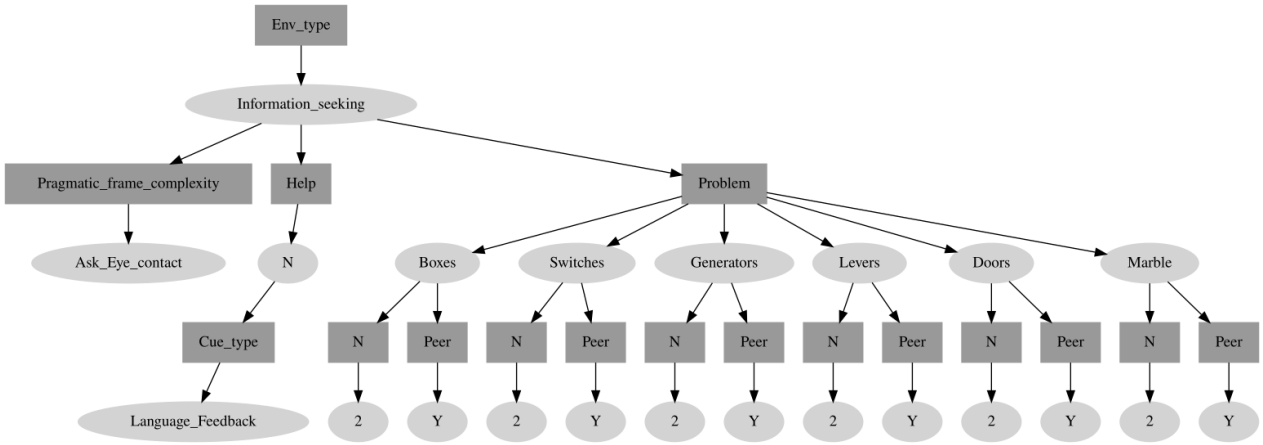


(d) Asocial group setting

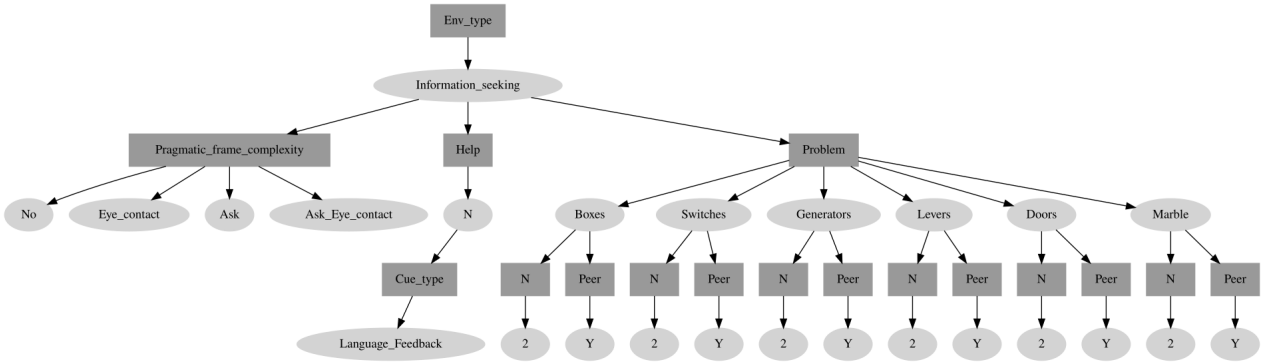


(e) Role B group setting

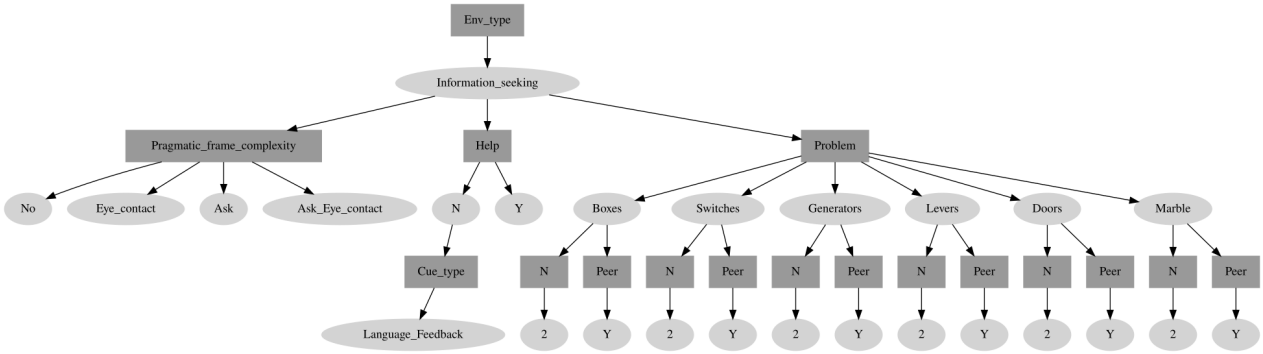
Figure 28. Role reversal sampling trees from the case study in section 5.2



(a) Testing tree.

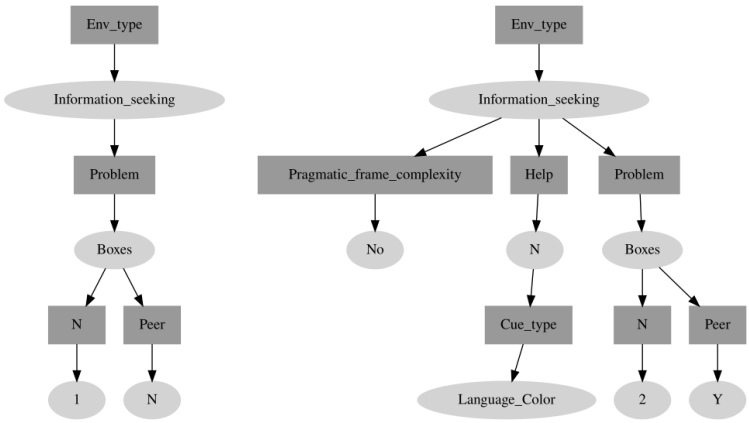


(b) Scaf_4 tree.



(c) Scaf_8 tree.

Figure 29. Sampling trees used in the first phase of the scaffolding case study in section 5.3



(a) Asocial Apple (b) Color boxes

Figure 30. Sampling trees used in the case study with LLMs as interactive agents (section 5.4)