A On Algorithmic Details of Meta-Referential Games

In this section, we detail algorithmically how Meta-Referential Games differ from common RGs. We start by presenting in Algorithm 4 an overview of the common RGs, taking place inside a common supervised learning loop, and we highlight the following:

(i) preparation of the data on which the referential game is played (highlighted in green),

(ii) elements pertaining to playing a RG (highlighted in blue),

(iii) elements pertaining to the **supervised learning loop** (highlighted in purple).

Helper functions are detailed in Algorithm 1, 2 and 3. Next, we can now show in greater and contrastive details the Meta-Referential Game algorithm in Algorithm 5, where we highlight the following:

(i) preparation of the data on which the referential game is played (highlighted in green),

(ii) elements pertaining to playing a RG (highlighted in blue),

(iii) elements pertaining to the **meta-learning loop** (highlighted in purple).

(iv) elements pertaining to setup of a Meta-Referential Game (highlighted in red).

Algorithm 1: Helper function : DataPrep

Given :

- a target stimuli s_0,
- a dataset of stimuli Dataset,
- O : Number of Object-Centric samples in each Target Distribution over stimuli $TD(\cdot)$.
- K : Number of distractor stimuli to provide to the listener agent.
- FullObs : Boolean defining whether the speaker agent has full (or partial) observation.
- DescrRatio : Descriptive ratio in the range $[0,1]$ defining how often the listener agent is observing the same semantic as the speaker agent.

```
$s'_0, D^{Target} \leftarrow s_0, 0$;
if $random(0,1) > DescrRatio$ then
    $s'_0 \sim$ Dataset $- TD(s_0)$; ;          /* Exclude target stimulus from listener's
      observation ...  */
    $D^{Target} \leftarrow K+1$; ;       /* ...  and expect it to decide accordingly.  */
end
else if $O > 1$ then
    Sample an Object-Centric distractor $s'_0 \sim TD(s_0)$;
end
Sample $K$ distractor stimuli from Dataset $- TD(s_0)$: $(s_i)_{i\in[1,K]} \sim$ Dataset $- TD(s_0)$;
$Obs_{\text{Speaker}} \leftarrow \{s_0\}$; if $FullObs$ then
    $Obs_{\text{Speaker}} \leftarrow \{s_0\} \cup \{s_i | \forall i \in [1,K]\}$;
end
$Obs_{\text{Listener}} \leftarrow \{s'_0\} \cup \{s_i | \forall i \in [1,K]\}$;
/* Shuffle listener observations and update index of target decision:
   */
$Obs_{\text{Listener}}, D^{Target} \leftarrow Shuffle(Obs_{\text{Listener}}, D^{Target})$;
```

Output : $Obs_{\text{Speaker}}, Obs_{\text{Listener}}, D^{Target}$;

Algorithm 2: Helper function : MetaRGDatasetPreparation

```
Given :
    • $V$ : Vocabulary (finite set of tokens available),
    • $N_{\text{dim}}$ : Number of attribute/factor dimensions in the symbolic spaces,
    • $V_{min}$ : Minimum number of possible values on each attribute/factor dimensions in the symbolic spaces,
    • $V_{max}$ : Maximum number of possible values on each attribute/factor dimensions in the symbolic spaces,
Initialise random permutation of vocabulary: $V' \leftarrow RandomPerm(V)$
Sample semantic structure: $(d(i))_{i \in [1, N_{\text{dim}}]} \sim \mathcal{U}(V_{min}; V_{max})^{N_{\text{dim}}}$;
Generate symbolic space/dataset $D((d(i))_{i \in [1, N_{\text{dim}}]})$;
Split dataset into supporting set $D^{\text{support}}$ and querying set $D^{\text{query}}$ $(((d(i))_{i \in [1, N_{\text{dim}}]})$ is omitted for
  readability);
Output : $V', D((d(i))_{i \in [1, N_{\text{dim}}]}), D^{\text{support}}, D^{\text{query}}$;
```

Algorithm 3: Helper function : PlayRG

```
Given :
    • Speaker and Listener agents,
    • Set of speaker observations $Obs_{\text{Speaker}}$,
    • Set of listener observations $Obs_{\text{Listener}}$,
    • $N$ : Number of communication rounds to play,
    • $L$ : Maximum length of each message,
    • $V$ : Vocabulary (finite set of tokens available),
Compute message $M^S$ = Speaker$(Obs_{\text{Speaker}}|\emptyset)$;
Initialise Communication Channel History: CommH $\leftarrow [M^S]$;
for $round = 0, N$ do
    Compute Listener's reply $M^L_{\text{round}}, \_ =$ Listener$(Obs_{\text{Listener}}|$CommH$)$;
    CommH $\leftarrow$ CommH $+ [M^L_{\text{round}}]$;
    Compute Speaker's reply $M^S_{\text{round}} =$ Speaker$(Obs_{\text{Speaker}}|$CommH$)$;
    CommH $\leftarrow$ CommH $+ [M^S_{\text{round}}]$;
end
Compute listener decision $\_, D^L =$ Listener$(Obs_{\text{Listener}}|$CommH$)$;
Output : Listener's decision $D^L$, Communication Channel History CommH;
```

Algorithm 4: Common Referential Game inside a Common Supervised Learning Loop

```
Given :
    • a dataset of stimuli $Dataset$,
    • a set of hyperparameters defining the RG:
        – $O$ : Number of Object-Centric samples in each Target Distribution over stimuli $TD(\cdot)$.
        – $N$ : Number of communication rounds to play.
        – $L$ : Maximum length of each message.
        – $V$ : Vocabulary (finite set of tokens available).
        – $K$ : Number of distractor stimuli to provide to the listener agent.
        – FullObs : Boolean defining whether the speaker agent has full (or partial) observation.
        – DescrRatio : Descriptive ratio in the range $[0, 1]$ defining how often the listener agent
          is observing the same semantic as the speaker agent.
        – $\mathcal{L}$ : Loss function to use in the agents update.
Initialize :
    • Speaker$(\cdot)$ and Listener$(\cdot)$ agents.
Systematically split $Dataset$ into training and testing dataset, $D^{\text{train}}$ and $D^{\text{test}}$;
for $epoch = 1, N_{epoch}$ do
    for target stimulus $s_0 \in D^{train}$ do
        /* Preparation of observations and target decision:                    */
        $Obs_{\text{Speaker}}, Obs_{\text{Listener}}, D^{Target} \leftarrow DataPrep(\text{Dataset}, s_0, O, K, \text{FullObs}, \text{DescrRatio})$
        /* Play Referential Game:                                               */
        $D^L, \_ = \text{PlayRG}(\text{Speaker}, \text{Listener}, Obs_{\text{Speaker}}, Obs_{\text{Listener}}, N, L, V)$;
        /* Supervised Learning Parameters Update on Training Stimulus Only:
           */
        Update both speaker and listener agents' parameters using the loss $\mathcal{L}(D^{Target}, D^L)$;
    end
    Initialise ZSCT accuracy: $Acc_{\text{ZSCT}} \leftarrow 0$;
    for target stimulus $s_0 \in D^{test}$ do
        /* Preparation of observations and target decision:                    */
        $Obs_{\text{Speaker}}, Obs_{\text{Listener}}, D^{Target} \leftarrow DataPrep(\text{Dataset}, s_0, O, K, \text{FullObs}, \text{DescrRatio})$
        /* Play Referential Game:                                               */
        $D^L, \_ = \text{PlayRG}(\text{Speaker}, \text{Listener}, Obs_{\text{Speaker}}, Obs_{\text{Listener}}, N, L, V)$;
        /* Update ZSCT Accuracy:                                                */
        $Acc_{\text{ZSCT}} \leftarrow \text{Update}(Acc_{\text{ZSCT}}, D^{Target}, D^L)$;
    end
end
```

Algorithm 5: Meta-Referential Game inside a Meta-Learning Loop

Given :

- $N_{episode}$, N_{dim} : Number of episodes, and number of attribute/factor dimensions,
- S : Minimum number of Shots over which each possible value on each attribute/factor dimension ought to be observed by the agents (as part of a target stimulus).
- V_{min}, V_{max} : Minimum and maximum number of possible values on each attribute/factor dimensions in the symbolic spaces,
- $TSS(\mathcal{D}, \mathcal{S}, S)$: Target stimulus sampling function which samples from dataset $\mathcal{D}$, given a set of previously sampled stimuli $\mathcal{S}$, while maximising the likelihood that each possible value on each attribute/factor dimension are sampled at least S times.
- a set of hyperparameters defining the RG:
  - O : Number of Object-Centric samples in each Target Distribution over stimuli $TD(\cdot)$.
  - N : Number of communication rounds to play.
  - L : Maximum length of each message.
  - V : Vocabulary (finite set of tokens available).
  - K : Number of distractor stimuli to provide to the listener agent.
  - FullObs : Boolean defining whether the speaker agent has full (or partial) observation.
  - DescrRatio : Descriptive ratio in the range $[0, 1]$ defining how often the listener agent is observing the same semantic as the speaker agent.

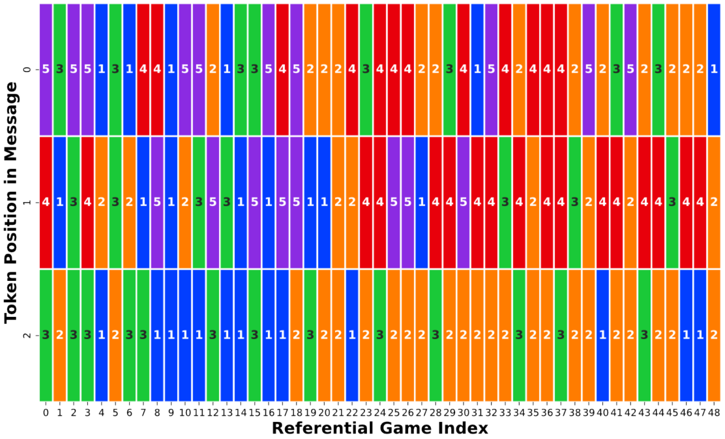
Initialize :

- Speaker$(\cdot)$ and Listener$(\cdot)$ agents.

```
for $episode = 1, N_{episode}$ do
    /* Preparation of the symbolic space/dataset:                          */
    $V', D_{\text{episode}}, D^{\text{support}}_{\text{episode}}, D^{\text{query}}_{\text{episode}} \leftarrow MetaRGDatasetPreparation(V, N_{\text{dim}}, V_{\text{min}}, V_{\text{max}})$;
    Initialise set of sampled supporting stimuli: $\mathcal{S}^{\text{support}} \leftarrow \emptyset$;
    repeat
        Sample training-purposed target stimulus $s^i_0 \sim TSS(D^{\text{support}}_{\text{episode}}, \mathcal{S}^{\text{support}}, S)$
        $\mathcal{S}^{\text{support}} \leftarrow \mathcal{S}^{\text{support}} \cup \{s^i_0\}$; $i \leftarrow i + 1$;
    until all values on each attribute/factor dimension have been instantiated at least $S$ times;
    Initialise RG index: $i \leftarrow 0$;
    /* Supporting Phase:                                                   */
    for target stimulus $s^i_0 \in \mathcal{S}^{support}$ do
        $Obs^i_{\text{Speaker}}, Obs^i_{\text{Listener}}, D^{Target}_i \leftarrow DataPrep(D^{\text{support}}_{\text{episode}}, s^i_0, O, K, \text{FullObs}, \text{DescrRatio})$;

        $D^L_i, CommH_i = \text{PlayRG}(\text{Speaker}, \text{Listener}, Obs^i_{\text{Speaker}}, Obs^i_{\text{Listener}}, N, L, V')$;
        $\_, \_ = \text{Listener}(Obs^i_{Speaker} | CommH_i)$ ;          /* Listener-Feedback Step */
    end
    /* Querying/ZSCT Phase:                                                */
    Initialise ZSCT accuracy: $Acc_{\text{ZSCT}} \leftarrow 0$;
    for target stimulus $s^i_0 \in D^{query}_{episode}$ do
        $Obs^i_{\text{Speaker}}, Obs^i_{\text{Listener}}, D^{Target}_i \leftarrow DataPrep(D_{\text{episode}}, s^i_0, O, K, \text{FullObs}, \text{DescrRatio})$;

        $D^L_i, CommH_i = \text{PlayRG}(\text{Speaker}, \text{Listener}, Obs^i_{\text{Speaker}}, Obs^i_{\text{Listener}}, N, L, V')$;
        $\_, \_ = \text{Listener}(Obs^i_{Speaker} | CommH_i)$ ;          /* Listener-Feedback Step */
        /* Update ZSCT Accuracy:                                           */
        $Acc_{\text{ZSCT}} \leftarrow \text{Update}(Acc_{\text{ZSCT}}, D^{Target}_i, D^L_i)$; $i \leftarrow i + 1$;
    end
    /* Meta-Learning Parameters Update on Whole Episode:                   */
    Update both agents using rewards $R_i = \begin{cases} 1 & \text{if } D^{Target}_i == D^L_i \\ 0 & \text{otherwise, during supporting phase}; \\ -2 & \text{otherwise, during querying phase} \end{cases}$
end
```



	d(0)=4 / d(1)=2 / d(2)=3		d(0)=3 / d(1)=3 / d(2)=3	
Example Latent Stimuli	(4 2 3)	(3 1 2)	(3 1 2)	(2 1 1)
MHE Sample Representation	(0001 0100 0010)	(0010 1000 0100)	(001 100 010)	(010 100 100)
SCS Sample Representation	$\begin{pmatrix} 0.82 \\ 0.55 \\ 0.74 \end{pmatrix}$	$\begin{pmatrix} 0.25 \\ -0.62 \\ 0.04 \end{pmatrix}$	$\begin{pmatrix} 0.95 \\ -0.90 \\ 0.10 \end{pmatrix}$	$\begin{pmatrix} -0.10 \\ -0.85 \\ -0.74 \end{pmatrix}$

Figure 4: **Top:** visualisation on each column of the messages sent by the posdis-compositional rule-based speaker agent over the course of the episode presented in Figure 3. Colours are encoding the information of the token index, as a visual cue. **Bottom:** OHE/MHE and SCS representations of example latent stimuli for two differently-structured symbolic spaces with $N_{dim} = 3$, i.e. on the left for $d(0) = 4$, $d(1) = 2$, $d(2) = 3$, and on the right for $d(0) = 3$, $d(1) = 3$, $d(2) = 3$. Note the shape invariance property of the SCS representation, as its shape remains unchanged by the change in semantic structure of the symbolic space, on the contrary to the OHE/MHE representations.

B Agent architecture & training

The baseline RL agents that we consider use a 3-layer fully-connected network with 512, 256, and finally 128 hidden units, with ReLU activations, with the stimulus being fed as input. The output is then concatenated with the message coming from the other agent in a OHE/MHE representation, mainly, as well as all other information necessary for the agent to identify the current step, i.e. the previous reward value (either $+1$ and 0 during the training phase or $+1$ and -2 during testing phase), its previous action in one-hot encoding, an OHE/MHE-represented index of the communication round (out of N possible values), an OHE/MHE-represented index of the agent's role (speaker or listener) in the current game, an OHE/MHE-represented index of the current phase (either 'training' or 'testing'), an OHE/MHE representation of the previous RG's result (either success or failure), the previous RG's reward, and an OHE/MHE mask over the action space, clarifying which actions are available to the agent in the current step. The resulting concatenated vector is processed by another 3-layer fully-connected network with 512, 256, and 256 hidden units, and ReLU activations, and then fed to the core memory module, which is here a 2-layers LSTM [Hochreiter and Schmidhuber, 1997] with 256 and 128 hidden units, which feeds into the advantage and value heads of a 1-layer dueling network [Wang et al., 2016].

Table 5 highlights the hyperparameters used for the learning agent architecture and the learning algorithm, R2D2[Kapturowski et al., 2018]. More details can be found, for reproducibility purposes, in our open-source implementation at HIDDEN_FOR_REVIEW_PURPOSE.

Training was performed for each run on 1 NVIDIA GTX1080 Ti, and the average amount of training time for a run is 18 hours for LSTM-based models, 40 hours for ESBN-based models, and 52 hours for DCEM-based models.

B.1 ESBN & DCEM

The ESBN-based and DCEM-based models that we consider have the same architectures and parameters than in their respective original work from Webb et al. [2020] and Hill et al. [2020], with the exception of the stimuli encoding networks, which are similar to the LSTM-based model.

B.2 Rule-based speaker agent

The rule-based speaker agents used in the single-agent task, where only the listener agent is a learning agent, speaks a compositional language in the sense of the posdis metric [Chaabouni et al., 2020], as presented in Table 4 for $N_{dim} = 3$, a maximum sentence length of $L = 4$, and vocabulary size $|V| >= max_i d(i) = 5$, assuming a semantical space such that $\forall i \in [1, 3], d(i) = 5$.

C Cheating language

The agents can develop a cheating language, cheating in the sense that it could be episode/task-invariant (and thus semantic structure invariant). This emerging cheating language would encode the continuous values of the SCS representation like an analog-to-digital converter would, by mapping a fine-enough partition of the $[-1, +1]$ range onto the vocabulary in a bijective fashion.

For instance, for a vocabulary size $\|V\| = 10$, each symbol can be unequivocally mapped onto $\frac{2}{10}$-th increments over $[-1, +1]$, and, by communicating N_{dim} symbols (assuming $N_{dim} \leq L$), the speaker agents can communicate to the listener the (digitized) continuous value on each dimension i of the SCS-represented stimulus. If $max_j d(j) \leq \|V\|$ then the cheating language is expressive-enough for the speaker agent to digitize all possible stimulus without solving the binding problem, i.e. without inferring the semantic structure. Similarly, it is expressive-enough for the listener agent to convert the spoken utterances to continuous/analog-like values over the $[-1, +1]$ range, thus enabling the listener agent to skirt the binding problem when trying to discriminate the target stimulus from the different stimuli it observes.

Table 4: Examples of the latent stimulus to language utterance mapping of the posdis-compositional rule-based speaker agent. Note that token 0 is the EoS token.

Latent Dims			Comp. Language
#1	#2	#3	Tokens
0	1	2	1, 2, 3, 0
1	3	4	2, 4, 5, 0
2	5	0	3, 6, 1, 0
3	1	2	4, 2, 3, 0
4	3	4	5, 4, 5, 0

D Further experiments:

D.1 On the BP instantiated by the SCS representation

Hypothesis. The SCS representation differs from the OHE/MHE one primarily in terms of the binding problem [Greff et al., 2020] that the former instantiates while the latter does not. Indeed, the semantic structure can only be inferred after observing multiple SCS-represented stimuli. We hypothesised that it is via the *dynamic binding of information* extracted from each observations that an estimation of a density distribution over each dimension i's $[-1, +1]$ range can be performed. And, estimating such density distribution is tantamount to estimating the number of likely gaussian distributions that partition each $[-1, +1]$ range.

Evaluation. Towards highlighting that there is a binding problem taking place, we show results of baseline RL agents (similar to main experiments in Section 4) evaluated on a simple single-agent recall task. The Recall task structure borrows from few-shot learning tasks as it presents over 2 shots all the stimuli of the instantiated symbolic space (not to be confused with the case for Meta-RG where all the latent/factor dimensions' values are being presented over S shots – Meta-RGs do not necessarily sample the whole instantiated symbolic space at each episode, but the Recall task does). Each shot consists of a series of recall games, one for each stimulus that can be sampled from an $N_{dim} = 3$-dimensioned symbolic space. The semantic structure $(d(i))_{i \in [1;N_{dim}]}$ of the symbolic space is randomly sampled at the beginning of each episode, i.e. $d(i) \sim \mathcal{U}(2; 5)$, where $\mathcal{U}(2; 5)$ is the uniform discrete distribution over the integers in $[2; 5]$, and the number of object-centric samples is $O = 1$, in order to remove any confounder from object-centrism.

Each recall game consists of two steps: in the first step, a stimulus is presented to the RL agent, and only a *no-operation* (NO-OP) action is made available, while, on the second step, the agent is asked to infer/recall the **discrete** $l(i)$ **latent value** (as opposed to the representation of it that it observed, either in the SCS or OHE/MHE form) that the previously-presented stimulus had instantiated, on a given i-th dimension, where value i for the current game is uniformly sampled from $\mathcal{U}(1; N_{dim})$ at the beginning of each game. The value of i is communicated to the agent via the observation on this second step of different stimulus that in the first step: it is a zeroed out stimulus with the exception of a 1 on the i-th dimension on which the inference/recall must be performed when using SCS representation, or over all the OHE/MHE dimensions that can encode a value for the i-th latent factor/attribute when using the OHE/MHE representation. On the second step, the agent's available action space now consists of discrete actions over the range $[1; max_j d(j)]$, where $max_j d(j)$ is a hyperparameter of the task representing the maximum number of latent values for any latent/factor dimension. In our experiments, $max_j d(j) = 5$. While the agent is rewarded at each game for recalling correctly, we only focus on the performance over the games of the second shot, i.e. on the games where the agent has theoretically received enough information to infer the density distribution over each dimension i's $[-1, +1]$ range. Indeed, observing the whole symbolic space once (on the first shot) is sufficient (albeit not necessary, specifically in the case of the OHE/MHE representation).

Results. Figure 5 details the recall accuracy over all the games of the second shot of our baseline RL agent throughout learning. There is a large gap of asymptotic performance depending on whether the Recall task is evaluated using OHE/MHE or SCS representations. We attribute the poor performance in the SCS context to the instantiation of a BP. We note again that during those experiments the number of object-centric samples was kept at $O = 1$, thus emphasising that the BP is solely depending on the use of the SCS representation and does not require object-centrism.

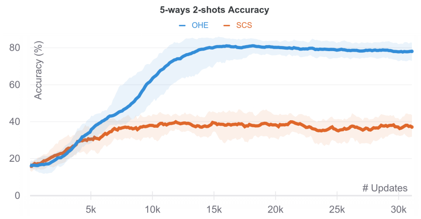


Figure 5: 5-ways 2-shots accuracies on the Recall task with different stimulus representation (OHE:blue ; SCS; orange).

D.2 On the ideally-disentangled-ness of the SCS representation

In this section, we verify our hypothesis that the SCS representation yields ideally-disentangled stimuli. We report on the **FactorVAE Score** Kim and Mnih [2018], the Mutual Information Gap (**MIG**) Chen et al. [2018], and the **Modularity Score** Ridgeway and Mozer [2018] as they have been shown to be part of the metrics that correlate the least among each other [Locatello et al., 2020], thus representing different desiderata/definitions for disentanglement. We report on the $N_{dim} = 3$-dimensioned symbolic spaces with $\forall j, d(j) = 5$ and $O = 5$. The measurements are of 100.0%, 94.8, and 98.9% for, respectivily, the FactorVAE Score, the MIG, and the Modularity Score, thus validating our design hypothesis about the SCS representation. We remark that the MIG and Modularity Score are sensitive to the number of object-centric samples O, which can be seen decreasing the measurements as low as 64.4% and 66.6% for $O = 1$. The FactorVAE Score is not affected, possibly due to its reliance on a deterministic classifier.

D.3 Auxiliary Reconstruction Loss

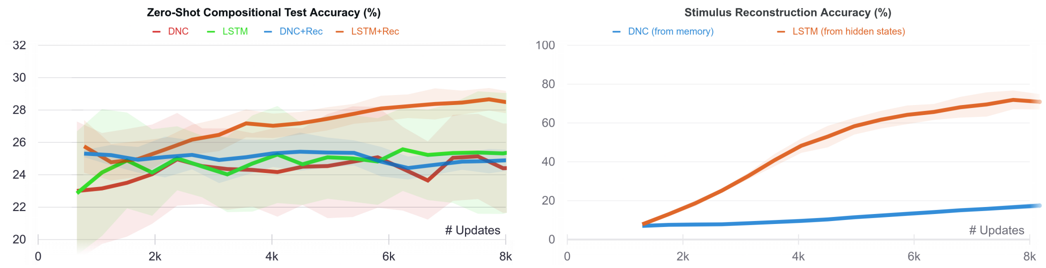
In the following, we investigate and compare the performance when using an LSTM [Hochreiter and Schmidhuber, 1997] or a Differentiable Neural Computer (DNC) [Graves et al., 2016] as core memory module, with or without the auxiliary reconstruction loss inspired from Hill et al. [2020].

In the case of the LSTM, the prediction network of the reconstruction loss takes as input the LSTM hidden states, while in the case of the DNC, the input is the memory. Figure 6b shows the stimulus reconstruction accuracies for both architectures, highlighting a greater data-efficiency (and resulting asymptotic performance in the current observation budget) of the LSTM-based architecture, compared to the DNC-based one.

Figure 6a shows the 4-ways (3 distractors descriptive meta-RGs) ZSCT accuracies of the different agents throughout learning. The ZSCT accuracy is the accuracy over querying-/testing-purpose stimuli only, after the agent has observed for two consecutive times (i.e. $S = 2$) the supportive training-purpose stimuli for the current episode. The DNC-based architecture has difficulty learning how to use its memory, even with the use of the auxiliary reconstruction loss, and therefore it utterly fails to reach better-than-chance ZSCT accuracies. On the otherhand, the LSTM-based architecture is fairly successful on the auxiliary reconstruction task, but it is not sufficient for training on the main task to really take-off. As expected from the fact that the benchmark instantiates a binding problem that requires relational responding, our results hint at the fact that the ability to use memory towards deriving valuable relations between stimuli seen at different time-steps is primordial. Indeed, only the agent that has the ability to use its memory element towards recalling stimuli starts to perform at a better-than-chance level. Thus, the auxiliary reconstruction loss is an important element to drive some success on the task, but it is also clearly not sufficient, and the rather poor results that we achieved using these baseline agents indicates that new inductive biases must be investigated to be able to solve the problem posed in our proposed benchmark.

E Broader impact

No technology is safe from being used for malicious purposes, which equally applies to our research. However, aiming to develop artificial agents that relies on the same symbolic behaviours and the same social assumptions (e.g. using CLBs) than human beings is aiming to reduce misunderstanding between human and machines. Thus, the current work is targeting benevolent applications. Subsequent works around the benchmark that we propose are prompted to focus on emerging protocols in general (not just posdis-compositional languages), while still aiming to provide a better understanding of artificial agent's symbolic behaviour biases and differences, especially when compared to human beings, thus aiming to guard against possible misunderstandings and misaligned behaviours. The current state of this work does not allow discussion of potential negative societal impact.



(a)

(b)

Figure 6: **(a):** 4-ways (3 distractors) zero-shot compositional test accuracies of different architectures. 5 seeds for architectures with DNC and LSTM, and 2 seeds for runs with DNC+Rec and LSTM+Rec, where the auxiliary reconstruction loss is used. **(b):** Stimulus reconstruction accuracies for the architectures augmented with the auxiliary reconstruction task. Accuracies are computed on binary values corresponding to each stimulus' latent dimension's reconstructed value being close enough to the ground truth value, with a threshold of 0.05 on each dimension, which correspond to a deviation tolerance of 2.5% since the range in which SCS stimuli are instantiated is $[-1, 1]$.

Table 5: Hyper-parameters values used in R2D2, with LSTM or DNC as the core memory module. All missing parameters follow the ones in Ape-X [Horgan et al., 2018].

R2D2	
Number of actors	32
Actor parameter update interval	1 environment step
Sequence unroll length	20
Sequence length overlap	10
Sequence burn-in length	10
N-steps return	3
Replay buffer size	5×10^4 observations
Priority exponent	0.9
Importance sampling exponent	0.6
Discount γ	0.997
Minibatch size	32
Optimizer	Adam [Kingma and Ba, 2014]
Optimizer settings	learning rate $= 6.25 \times 10^{-5}$, $\epsilon = 10^{-12}$
Target network update interval	2500 updates
Value function rescaling	None

Core Memory Module			
LSTM [Hochreiter and Schmidhuber, 1997]		DNC [Graves et al., 2016]	
Number of layers	2	LSTM-controller settings	2 hidden layers of size 128
Hidden layer size	256, 128	Memory settings	128 slots of size 32
Activation function	ReLU	Read/write heads	2 reading ; 1 writing