LETTER Communicated by Hans Briegel

Enhanced Equivalence Projective Simulation: A Framework for Modeling Formation of Stimulus Equivalence Classes

Asieh Abolpou Mofrad
asieh.abolpour-mofrad@oslomet.no
Anis Yazidi
Anis.Yazidi@oslomet.no
Department of Computer Science, Oslo Metropolitan University,
0130 Oslo, Norway

Samaneh Abolpour Mofrad
Samaneh.Abolpour.Mofrad@hvl.no
Department of Computer Science, Electrical Engineering, and Mathematical Sciences, Western Norway University of Applied Sciences, 5063 Bergen, Norway, and Mohn Medical Imaging and Visualization Center, Department of Radiology, Haukeland University Hospital, 5021 Bergen, Norway

Hugo L. Hammer
Hugo.Hammer@oslomet.no
Department of Computer Science, Oslo Metropolitan University, 0130 Oslo, Norway, and Simula Metropolitan Center, 1325 Oslo, Norway

Erik Arntzen
erik.arntzen@equivalence.net
Department of Behavioral Science, Oslo Metropolitan University,
0130 Oslo, Norway

Formation of stimulus equivalence classes has been recently modeled through equivalence projective simulation (EPS), a modified version of a projective simulation (PS) learning agent. PS is endowed with an episodic memory that resembles the internal representation in the brain and the concept of cognitive maps. PS flexibility and interpretability enable the EPS model and, consequently the model we explore in this letter, to simulate a broad range of behaviors in matching-to-sample experiments. The episodic memory, the basis for agent decision making, is formed during the training phase. Derived relations in the EPS model that are not trained directly but can be established via the network's connections are computed on demand during the test phase trials by likelihood reasoning. In this letter, we investigate the formation of derived relations in the EPS model using network enhancement (NE), an iterative diffusion process, that yields an offline approach to the agent decision

Neural Computation **33, 483–527** (2021)
https://doi.org/10.1162/neco_a_01346

making at the testing phase. The NE process is applied after the training phase to denoise the memory network so that derived relations are formed in the memory network and retrieved during the testing phase. During the NE phase, indirect relations are enhanced, and the structure of episodic memory changes. This approach can also be interpreted as the agent's replay after the training phase, which is in line with recent findings in behavioral and neuroscience studies. In comparison with EPS, our model is able to model the formation of derived relations and other features such as the nodal effect in a more intrinsic manner. Decision making in the test phase is not an ad hoc computational method, but rather a retrieval and update process of the cached relations from the memory network based on the test trial. In order to study the role of parameters on agent performance, the proposed model is simulated and the results discussed through various experimental settings.

1 Introduction

Stimulus equivalence (SE), a phenomenon that Sidman (1971) identified and explored, refers to the condition that members of an equivalence class evoke the same response in human and animal subjects. The SE methodology uses a matching-to-sample (MTS) procedure to train arbitrary relations between unfamiliar stimuli and test derived relations through mathematical relations in equivalence sets: reflexivity, symmetry, and transitivity. The SE framework, as an efficient learning method, has been widely studied by employing humans or animals as experimental participants (see Sidman, Cresson, & Willson-Morris, 1974; Sidman et al., 1982; Sidman & Tailby, 1982; Sidman, Willson-Morris, & Kirk, 1986; Devany, Hayes, & Nelson, 1986; Hayes, 1989; Fields, Adams, Verhave, & Newman, 1990; Spencer & Chase, 1996; Groskreutz, Karsina, Miguel, & Groskreutz, 2010; Steingrimsdottir & Arntzen, 2011; Arntzen & Mensah, 2020, to mention a few). Computational models constitute another alternative for understanding SE and studying variables that are challenging to examine on humans or animals due to time constraints or ethical issues (see, e.g., Barnes & Hampson, 1993; Cullinan, Barnes, Hampson, & Lyddy, 1994; Lyddy, Barnes-Holmes, & Hampson, 2001; Lew & Zanutto, 2011; Tovar & Westermann, 2017; Ninness, Ninness, Rumph, & Lawson, 2018, for some computational models of the learning of equivalence relations).

In our previous model (Mofrad, Yazidi, Hammer, & Arntzen, 2020), we proposed equivalence projective simulation (EPS) for computationally modeling the SE phenomenon. In brief, EPS has modeled the formation of SE classes through an MTS procedure. A projective simulation (PS) framework (Briegel & De las Cuevas, 2012) was the basis of the model, and we have proposed several methods to address the test phase and derived relations, including max-product, memory sharpness, and random walk on the memory network with absorbing action sets. The EPS model, similar to the original PS model, has an internal episodic memory that is updated during the training phase, which is used to cope with new, derived relations in the testing phase. The PS model, and therefore the EPS model, is flexible and easy to interpret, which allows modeling a broad range of behaviors in MTS experiments, including typical participants or participants with some disabilities. Many parameters of the model can be controlled, such as the learning rate, forgetting rate, and nodal effect.

The EPS model relies on the assumption that the relations are derived on request, that is, when they appear in an MTS trial during the testing phase and updated during the training phase. We slightly change this assumption and form those relations at the end of the training phase; thus, the output network from the training phase of EPS is assumed to be a noisy version of the agent's memory network that is supposed to contain all trained and derived relations. Using a denoising approach, we could produce a new, less noisy clip network that contains information regarding equivalence class formation. The trained relations in the training phase are mapped into a transition matrix whose values describe the strength of the trained relations. By resorting to network enhancement (Wang et al., 2018), we address the formation of SE classes using an iterative update of the transition matrix. Interestingly, the updating process permits naturally denoising the transition matrix and enhancing indirect relations[1] while preserving the initial direct relations learned during the training phase. The denoised network can be assimilated to an updated clip network, used later in the testing phase. It can also be used to assess overall agent performance on eventual equivalence tests. In summary, the contribution of this letter is as follows:

1. Instead of using reasoning, that is, computing the likelihood of the different alternatives during testing by following some indirect paths over the clip network, we update memory and retrieve the updated memory at the testing phase.
2. As in the EPS model, we still control symmetry relations with a multiplicative parameter. We are able to control the ability to derive transitivity relations using parameter α. This turns out to be of great importance when modeling subjects with learning disabilities.
3. We further enhance the NE and propose DNE in which we can control the agent's ability to derive symmetry and also control its ability to derive transitivity.
4. A comparison of PS, EPS, and E-EPS, together with supporting studies from the neuroscience literature, is provided that justifies the proposed model.

[1] According to the theory of SE, indirect relations are derived through reflexivity, symmetry, transitivity, and equivalence.

5. From a computational point of view, the new updating rule has fewer parameters to fine-tune in comparison with the EPS. The approach to deriving relations in EPS model can be seen as routing in the clip network, with action sets as destination points. In the E-EPS model, a diffusion model explores the clip network by simultaneous propagation of flow without a specific target.
6. The updated clip network can be considered as a cognitive map of stimuli that can be used in analyzing the results of different settings.
7. The testing phase in the E-EPS model involves less computation on the decision time in comparison with EPS. E-EPS uses the updated network during the testing phase rather than processing the trained relations to compute derived relation links at each test trial.
8. Using a simulation of several configurations, we study the parameters in detail.
9. We compare three training procedures—linear series (LS), many-to-one (MTO), and one-to-many (OTM)—in the final experiment. In line with the mainstream literature in behavior analysis (see Arntzen, Grondahl, & Eilifsen, 2010; Arntzen & Hansen, 2011; Arntzen, 2012), the model yields better performance in OTM and MTO cases in comparison with LS, which is a qualitative property of our model confirming that it is a realistic model.
10. We provide theoretical analysis of the model and a convergence guarantee in appendix A.

We provide a brief overview of SE, EPS, and network enhancement in section 2. We provide the architecture of the enhanced equivalence projective simulation (E-EPS) model in section 3, where we also compare the proposed approach to the original PS model and recent EPS model. We consider seven experimental scenarios to study the parameters of the model in section 4. Section 5 offers a summary of the letter discussion, and concluding remarks.

2 Background and Related Work

In section 2.1, we review the concept of SE from a behavior analysis perspective. In section 2.2, we briefly explain the EPS model and provide a brief section about network enhancement (Wang et al., 2018) in section 2.3.

2.1 Stimulus Equivalence (SE).

SE is a research method on complex human behavior, including memory and problem solving (Sidman, 1990). In the MTS or conditional discrimination procedure, which is used in SE, a given stimulus, say A_1, must be paired with B_1 among a given comparison stimuli set, say B_1, B_2, and B_3. The discrimination happens through programmed consequences.

The MTS procedure has two phases: the training phase, when the participant learns some relations, and the testing phase, when the participant is tested with derived relations. Trial types in the testing phase include baseline, symmetry, transitivity, and equivalence. It is noteworthy that equivalence relations are sometimes referred to as combined transitivity and symmetry.

The evaluation of participant learning is usually through a threshold or mastery criterion ratio (e.g., 0.95 to 1). If the participant passes the criterion, the derived relations are tested. In the testing phase, there are no programmed consequences, and usually the criterion ratio in this phase is lower than in training phase (e.g., 0.9 to 1). Whenever the evidence (passing the criterion for testing) shows the emergence of all relations, the equivalence class is considered to be formed (Sidman & Tailby, 1982).

In the equivalence literature, three training structures have been used in establishing conditional discrimination with the MTS procedure: linear series (LS), many-to-one (MTO), and one-to-many (OTM) (see Arntzen, 2012, for more details about MTS training and testing procedures and parameters in SE formation). Generally, a class with n stimuli, requires training of only $(n - 1)$ stimulus-stimulus relations. The condition is that each component of these relations needs to be present in at least one trained relation, and none of the trained relations can have the same two stimuli as components. Even with these constraints, many possible ways for structuring training relations remain, some of them possibly more efficient than the others (see Fields et al., 1990; O'Mara, 1991; Arntzen & Holth, 1997; Hove, 2003; Lyddy & Barnes-Holmes, 2007; Arntzen et al., 2010; Arntzen & Hansen, 2011; Fienup, Wright, & Fields, 2015, for instance). Appendix B formally analyzes the size of the training design space, which is shown to be exhaustive even for a small number of categories and number of classes. Therefore, it is complex to design and run experiments involving human subjects that explore different training and testing scenarios. Computational models, however, could be used for exploring new ideas through simulation. For instance, one could try several configurations and find the optimum scenario according to some design criterion in the computational model before running a real experiment. Moreover, components of the computational model can be easily manipulated, disrupted, impaired, and removed to see the effect of those components on the results. Having more control over the experimental variables, including a controllable environment, is a considerable advantage of these models over real experiments (Barnes & Hampson, 1993; McClelland, 2009; Ninness, Ninness, Rumph, & Lawson, 2018).

2.2 Equivalence Projective Simulation (EPS). EPS is based on PS, which can be seen as an reinforcement learning (RL) model that can be embodied in an environment, perceive stimuli, execute actions, and learn through trial and error (see, e.g., Briegel & De las Cuevas, 2012; Melnikov, Makmal, Dunjko, & Briegel, 2017, for details of PS model).

The PS agent, and therefore the EPS agent, has an episodic memory that is literally a directed, weighted network of clips, where each clip represents a remembered percept or action (stimulus in EPS). Memory can be described as a probabilistic network of clips, the so-called clip network.[2] The learning in PS is realized by updating weights and structure through adding new clips and new transition links.

The simulation of the MTS procedure via EPS has two phases: the training phase, when the memory network will be formed through trials and guided feedback, and the testing phase, when no new memory clips are created. Although there is no guided feedback in the testing phase, connection weights might be updated. The testing phase is the main part of the model. In Mofrad et al. (2020) three different approaches dealing with the derived relations are discussed: max-product, memory sharpness, and absorbing action sets.

At the beginning of an MTS training phase, the agent memory space, which is shown by $\mathcal{C} = \{c_1, \ldots, c_p\}$, is empty. Based on trial settings, a memorized clip could play the role of either a percept clip or an action clip. At each time step, the environment (the experimenter in the real experiments) shows a sample stimulus and some comparison alternatives, which are referred to as percept and actions. The percept and actions belong to the percept set $\mathcal{S}$ and action set $\mathcal{A}$, respectively. The sample stimulus (percept, $s \in \mathcal{S}$) and the comparison stimuli (actions $a \in \mathcal{A}_t$) belong to different categories (e.g., category A or B), where $\mathcal{A}_t$ denotes the action space at time t and consists of a set of comparisons at the given trial. The training phase will be as follows:

1. The agent perceives stimulus $s \in \mathcal{S}$ from the environment. Clip $c_s \in \mathcal{C}$ is either created (the first time) or activated.
2. Perceiving action set $\mathcal{A}_t$ from the environment, the agent establishes and initializes connections between the sample and comparison stimuli the first time with h-values equal to h_0. If there exist connections from previous trials, there is no need for initialization.
3. The agent computes $p^{(t)}(c_a|c_s)$, $a \in \mathcal{A}_t$ based on the h-values using the softmax distribution function,

$$p^{(t)}(c_j|c_i) = \frac{e^{\beta h^{(t)}(c_i,c_j)}}{\sum_k e^{\beta h^{(t)}(c_i,c_k)}}, \tag{2.1}$$

where at this stage, clip $c_i = c_s$ and clip $c_j \in \mathcal{A}_t$. A larger value of $\beta \geq 0$ creates a probability distribution that is more biased to the choice of

[2] The terms *episode* and *clip* are used interchangeably.

the largest h-value, and therefore parameter β can be used for tuning the learning rate as well.

4. The agent selects one of the actions based on the computed probability distribution and receives a positive or negative reward from the environment, say, $\lambda^{(t)} \in \Lambda = \{-1, 1\}$.[3]
5. The connection weights, h-values, will be updated as a result of the environment feedback as follows:

$$h^{(t+1)}(c_s, c_a) = h^{(t)}(c_s, c_a) - \gamma(h^{(t)}(c_s, c_a) - 1) + \lambda^{(t)}. \tag{2.2}$$

Moreover, the opposite link, (c_a, c_s), will be updated in a similar way, but with the parameter $0 < K \leq 1$:

$$h^{(t+1)}(c_a, c_s) = h^{(t)}(c_a, c_s) - \gamma(h^{(t)}(c_a, c_s) - 1) + K\lambda^{(t)}. \tag{2.3}$$

6. The environment provides new trials until all training relations meet the mastery criterion.

It is noteworthy that parameter K was used in the learning rule of the original PS model (Briegel & De las Cuevas, 2012) to determine the growth rate of associative or compositional connections relative to the direct connections. This parameter, for instance, enables the PS agent to learn faster by recognizing similarity among the existing clips in memory and new perceptual input (see Figures 11 and 12 in Briegel & De las Cuevas, 2012, for more detail on associative learning in the PS agent). The parameter K in the EPS model, however, quantifies the relative growth of symmetry relations compared to the direct, or baseline, relations.[4] This parameter is different from the original PS in the sense that the stimuli in EPS (and E-EPS) are arbitrary, that is, they have no physical similarity, and therefore the parameter K does not capture similarity. The notion of associative memory, however, can be added to the EPS model by introducing compound stimuli, which we do not address in this letter.

After that agent passes the training phase, the testing phase, in which the formation of derived relations is tested starts. At this stage, no feedback is provided from the environment.

1. The agent perceives $s \in \mathcal{S}$, activates the memory clip $c_s \in \mathcal{C}$, and tries to choose the best action among the given action set $\mathcal{A}_t$ based on its memory as follows.

[3] It is noteworthy that Λ could have any positive or negative values, including asymmetric rewards. For instance, negative feedback might have greater impact (see Baumeister, Bratslavsky, Finkenauer, & Vohs, 2001, as an example of a positive-negative asymmetry effect).

[4] In Mofrad et al. (2020), we use K_1, K_2, K_3, and K_4, which play the same role as K in this letter but with a higher level of control.

2. If connections between the sample and comparisons exist, the agent computes the $p^{(t)}(c_a|c_s), a \in \mathcal{A}_t$ based on the h-values using a probabilistic distribution achieved by either softmax or a normalized vector of h-values (called "standard" in PS and EPS). If such connections do not exist in the transitivity or equivalence relation cases, the agent computes the transition probabilities using a max-product scenario or an absorbing states scenario and selects one of the possible actions:
   - In the max-product case, the agent finds the most probable paths between c_s and each action $c_a, a \in \mathcal{A}_t$. There are many possible paths that might link c_s to a particular action c_a, and thus the procedure might be computationally exhaustive.
   - The absorbing state scenario can be considered as a random clip network, starting from c_s and ending with a clip in $\mathcal{A}_t$. So, unlike the max-product method, the probability of reaching each action from c_s is important but not the path itself. These probabilities can be computed when actions $c_a \in \mathcal{A}_t$ are set to be absorbing states of the underlying Markov chain at time t.
3. Memory sharpness, $0 \leq \theta \leq 1$, functions as a mechanism to control the formation of transitivity relations and consequently controls equivalence relations and the effect of the nodal number (see, e.g., Sidman, 1994, for nodal number), in line with the baseline relations training. Mofrad et al. (2020) discuss memory sharpness as a separate method. However, it can be used in combination with either max-product or the concept of absorbing states.

For the sake of brevity, we review just the parts of the EPS model that are necessary for understanding the new perspective on derived relations. Moreover, an overview of some other behavior-analytic computational approaches to the formation of SE classes is provided in Mofrad et al. (2020), which provides a detailed version of EPS model.

2.3 Network Enhancement (NE). Wang et al. (2018) proposed network enhancement (NE), a computational approach for denoising biological networks. NE converts a noisy, undirected, weighted network into a new network possessing the same nodes but with different connections and weights. It assumes that nodes that are connected through paths with high weight edges have a high chance of being directly connected with a high-weight edge. The NE diffusion process uses random walks of length 3 or less and a regularized information flow in order to produce new edge weights.

For a formal description of NE, let W be the matrix of edge weights and $\mathcal{N}_i$ be the K-nearest neighbors of the ith node, including node i itself. The localized network $\mathcal{T}$ is constructed from W as follows:

$$P_{i,j} \leftarrow \frac{W_{i,j}}{\sum_{k\in\mathcal{N}_i} W_{i,k}} \mathbb{I}_{\{j\in\mathcal{N}_i\}}, \;\; \mathcal{T}_{i,j} \leftarrow \sum_{k=1}^{n} \frac{P_{i,k}P_{j,k}}{\sum_{v=1}^{n} P_{v,k}}, \tag{2.4}$$

where $\mathbb{I}_{\{.\}}$ is the indicator function. Then the diffusion process is defined as an iterative relation,

$$W_{t+1} = \alpha\mathcal{T} \times W_t \times \mathcal{T} + (1-\alpha)\mathcal{T}, \tag{2.5}$$

where α is a regularization parameter, t shows the iteration step, and W_0 can be initialized with the input matrix W. The update rule in equation 2.5 for each entry is

$$(W_{t+1})_{i,j} = \alpha \sum_{k\in\mathcal{N}_i} \sum_{l\in\mathcal{N}_j} \mathcal{T}_{i,k}(W_t)_{k,l}\mathcal{T}_{l,j} + (1-\alpha)\mathcal{T}_{i,j}. \tag{2.6}$$

The many theoretical properties for this diffusion process are discussed in Wang et al. (2018). It is shown that W_t remains a symmetric, doubly stochastic matrix (DSM) for each iteration t, and W_t converges to a nontrivial equilibrium network. Moreover, NE does not change eigenvectors of the initial DSM $\mathcal{T}$, but the spectrum of the eigenvalues is changed nonlinearly so that the eigengap is increased. This effect of the NE process on the eigenspectrum improves the network to achieve a more accurate detection of clusters. Although this method produces promising results in our model, as we will explain in section 4, it is not the main approach for formation of equivalence classes in the EPS model, but NE and discussions in Wang et al. (2018) are the main motivation for the update rule. The method we use does not have all the properties that NE has, and we refer to the theoretical aspect of the diffusion process we used in appendix A. In the rest of this letter, we refer to the NE method due to Wang et al. (2018) as *symmetric network enhancement* (SNE).

3 Enhanced Equivalence Projective Simulation (E-EPS)

The training phase of the proposed E-EPS model is generally the same as the original PS and the EPS in the sense that the clip network is formed by adding new clips and updating the h-values based on the environment feedback. However, since in this letter, the probability distribution over the action set is modeled using the softmax function, we let the network have negative h-values and simplify the training by removing some parameters associated with positive h-values. However, the approach to the formation of SE classes and the testing phase is quite different compared to the EPS model (Mofrad et al., 2020). As we explained in section 2.2, after the training phase, we have a network of h-values for baseline relations and the symmetry relations. To add reflexivity to the clip network, we can consider an updating method either during the training phase[5] or after the training phase. In order to keep the model simple, we add a self-loop to each clip after the training phase and assign it an *h*-value equal to the maximum *h*-value of input or output connections. The argument is that in the case where the agent can identify the members of a class (say, A_1, B_1, C_1), it must be able to differentiate members of each category (say, A_1 from A_2 and A_3). We refer to the adjacency matrix of this network of *h*-values as W_h.

In this work, we are proposing a new NE model called *directed network enhancement* (DNE) that can be used for the testing phase, including baseline, reflexivity, symmetry, transitivity, and equivalence relations. Consider the following rule as the update rule (or diffusion process),

$$W_{t+1} = \alpha P \times W_t \times P + (1-\alpha)P, \tag{3.1}$$

where W_0 is a right stochastic matrix achieved from W_h. (By a "right stochastic matrix," we mean a real square matrix in which each row sums to one.) We put $W_0 = P$ where P is the transition probability matrix of W_h applying softmax function on nonzero values at each row using β_h parameter. P is not symmetric, and $P\mathbf{1} = \mathbf{1}$, where $\mathbf{1}$ represents the all-one eigenvector of P associated with eigenvalue one. In other words, P is a right stochastic matrix, so it can be used as the initial matrix in the DNE process. In the theoretical analysis of the SNE process provided by Wang et al. (2018), and supplementary note 3, the converged network is proved to be

$$W_{t\to\infty} = (1-\alpha)\mathcal{T}(\mathcal{I} - \alpha\mathcal{T}^2)^{-1}. \tag{3.2}$$

As we discuss in appendix A, the convergence in the DNE process remains valid for a network where we substitute $\mathcal{T}$ with P in equation 3.2:

$$W_{t\to\infty} = (1-\alpha)P(\mathcal{I} - \alpha P^2)^{-1}. \tag{3.3}$$

This post-processing phase transforms the *h*-value network obtained by training into a new network that can represent the agent predictive representations in a cognitive map (or successor representation similar to Momennejad, Russek et al., 2017).

The $W_{t\to\infty}$ matrix can be seen as the memory representation where we ignore the effect of context (or actions) and assume all the transitions in the network are based on the random walk on the graph (or diffusion). For instance, we can interpret the (i, j) entry of the $W_{t\to\infty}$ matrix as the transition probability from clip i to clip j when there is no external control.

[5] For instance, this can simply be achieved by adding a self-loop edge initialized with h_0 to each clip the first time it is perceived by the agent and update it whenever it gets involved in a trial.

When it comes to the testing phase, the softmax function with β_t is applied to calculate the probability distribution for each test trial. In order to accommodate the controlling effect of the test trials, the input values to the softmax function are set to be conditional probabilities given the trial, which can be calculated using Bayes' rule. As an example, if the test trial consists of A_1 as the sample stimulus and $F = \{F_1, F_2, F_3\}$ as the comparison stimuli, input values for the softmax function are $P(A_1F_1|A_1F)$, $P(A_1F_2|A_1F)$, and $P(A_1F_3|A_1F)$ where event A_1F is either A_1F_1, A_1F_2, or A_1F_3. These conditional values can be calculated due to Bayes' rule, for instance,

$$\begin{aligned}&P(A_1F_1|A_1F)\\ &\quad= \frac{P(A_1F_1)P(A_1F|A_1F_1)}{P(A_1F_1)P(A_1F|A_1F_1)+P(A_1F_2)P(A_1F|A_1F_2)+P(A_1F_3)P(A_1F|A_1F_3)}\\ &\quad= \frac{P(A_1F_1)}{P(A_1F_1)+P(A_1F_2)+P(A_1F_3)},\end{aligned}$$

which can be seen as a normalization. Note that all the conditional probabilities on the right-hand side are equal to one and therefore are removed. Parameter β_t in the softmax function can characterize the agent's memory and ability to link an internal representation to the real action. When a test trial is given to the agent, the memory is conditioned based on the test trials (sample and comparison stimuli), and Bayes' rule is used to characterize the environment effect.

Another way to formalize the behavior of the agent in the testing phase is to use a trial-based β_t for the softmax function, which is defined as β_t divided by the summation over weights for comparison stimuli. The above example, with A_1 as the sample stimulus and $F = \{F_1, F_2, F_3\}$ as the comparison stimuli, uses $\frac{\beta_t}{P(A_1F_1)+P(A_1F_2)+P(A_1F_3)}$ as the β in softmax function. As is clear from the example, in this formalization, the results remain exactly the same but they open up room to interpret the agent behavior differently. Using Bayes's rule and a fixed β_t approach emphasizes the effect of environment and the agent characteristics separately, but the variable β_t approach avoids the interpretation that the agent probabilities are calculated twice.

Before comparing the E-EPS to the original PS and the EPS model and relating it to other studies, we summarize the parameters of the agent model:

1. Parameter $0 < K \leq 1$ controls the formation of symmetry relations. $K = 1$ means that the relations are bidirectional and the h-value network is symmetric (see experiment 2).
2. Parameter $0 \leq \gamma < 1$ represents the forgetting rate during the training phase. The training structure (order of relations to be trained) is more important when the forgetting rate is high (see experiment 4).

3. Parameter $\beta_h > 0$ converts h-values to probabilities during the training trials and generates the input matrix W_0 for the NE process (see experiments 1 and 3).
4. Parameter $0 \leq \alpha < 1$ controls to what extent the NE affects the initially trained network when there is no test trial in place. α could characterize the amount of abstract mental process or replay that the agent performs. Even a small value of α could form derived relations that are weak compared to direct relations, but the ratio or conditional probabilities (used as an input to the softmax function) are strong. A value close to one for α means too much diffusion, which can erase the trained relations. One might find the appropriate diffusion based on the expected agent abilities and the training criterion (see experiment 5 and appendix A for more details)
5. Parameter $\beta_t > 0$ controls the agent's performance in a test trial (see experiment 6).

3.1 PS, EPS, and E-EPS: Discussion and Comparison. As Briegel and De las Cuevas (2012) mentioned, the idea of a clip network in PS is similar to the idea of Tolman's (1948) cognitive maps, which refers to a rich internal model of the world that represents relationships between events and simulates the consequences of actions. Although cognitive maps are mostly used for modeling spatial behavior (O'Keefe & Nadel, 1978), they are more general and cover the organization of knowledge in other types of behaviors, including flexible behavior. Cognitive maps can be constructed from abstract representations to describe relational knowledge, and new cognitive problems can then be considered as inference in this relational basis (Behrens et al., 2018).

Brain studies suggest multiple solutions to predicting long-term reward in RL problems (Daw, Niv, & Dayan, 2005). Learning a model of the environment, or a cognitive map of the environment, and using it to simulate future states step-by-step to predict long-term reward are different solutions, which we refer to as model-based RL (Daw et al., 2005; Daw, Gershman, Seymour, Dayan, & Dolan, 2011; Sutton & Barto, 2018). Forming simple world models in the human hippocampus for relational knowledge sorting and value spreading across associated stimulus representations is shown to directly influence behavior in a novel decision-making situations (Wimmer & Shohamy, 2012). Repeating patterns during both awake experiential states and nonengaged states and reshaping neural circuits has been studied in both the hippocampus and the neocortex (see Liu & Watson, 2020, for a review). Functional magnetic resonance imaging (fMRI) similarity measures in the hippocampus and entorhinal cortex (Stachenfeld, Botvinick, & Gershman, 2017; Garvert, Dolan, & Behrens, 2017) suggest the existence of statistical transitions of discrete state-spaces. The use of precompiled transition distances, rather than simulating all possible transitions online, is studied by Momennejad, Russek et al. (2017), where these precompiled distances depend on offline activity, or replay, in the hippocampus and ventral frontal cortex (Momennejad, Otto, Daw, & Norman, 2017). Caching of multistep predictive representations is also referred to as a "predictive map" (Stachenfeld et al., 2017). These predictive representations link model-based RL to model-free mechanisms through offline replay mechanisms (Russek, Momennejad, Botvinick, Gershman, & Daw, 2017) resembling Dyna-style planning (Sutton, Szepesvári, Geramifard, & Bowling, 2008).

PS is much more primitive than Dyna-style planning. It only changes the weights of the clip transition and performs a random walk on the clip network (for a detailed comparison, see Mautner, Makmal, Manzano, Tiersch, & Briegel, 2015). The multiple reflection in the PS model is different from "experiment replay" (Lin, 1992) in the sense that PS uses short-term memory, or emotional tags, to evaluate the result of a simulation and repeat the random walk if the remembered reward for the chosen action in the previous round was negative. So repeatedly presenting its experiences to its learning algorithm is not performed just for the sake of memory consolidation. (See also Momennejad, 2020, for a review on the role of replay on how the brain learns and generalizes relational structures with a focus on the RL approach.)

In the EPS model (Mofrad et al., 2020), two scenarios, called "standard" and "softmax," were used for the training phase, and various ways for deriving relations in the test phase were studied and discussed due to the aim to define EPS as a general and flexible model. The EPS (and E-EPS) training phase is similar to the PS model with extra links and update rule for symmetry relations. In this letter, we survey just the training method that uses the softmax function in order to calculate probability distributions over the action sets. Although the training phase in this letter could be similar to EPS, for simplicity, we just consider the softmax scenario where negative h-values are allowed, so we can formalize the learning with just one parameter, K, to control the growth ratio of symmetry relations in comparison with the direct relations.

The main difference with PS, the most important part of the EPS (E-EPS) model, is the testing phase where there is no feedback. In the EPS model, the derived relations were calculated on demand at the decision time whenever they appear in a test trial. The probabilities are either calculated based on the probabilities of the paths with maximum values, using a max-product algorithm, or the probability of reaching each of the action points having a random walk on the episodic memory started at a sample stimulus. The symmetry relations are controlled via a multiplicative parameter, and the transitivity can be controlled with a parameter called memory sharpness.

In the EPS testing phase, the only change to the clip network h-values is related to the parameter γ, the forgetting factor, and all the computations for the test trials are performed at the decision time, which can be seen as an ad hoc computational tool rather than an intrinsic feature of the model. The perspective to the derived relation in E-EPS, is quite different where NE, an iterative diffusion process, is used after the training phase. This alternative approach updates the structure of clip network by adding new connections between the clips and updating connection weights. In other words, the approach to derive relations in the EPS model can be seen as routing in the clip network, where the action sets play the role of destination, while in the E-EPS model, in the absence of test trials, the approach involves a diffusion model to explore the clip network by simultaneous propagation of flow without a specific target. The NE process is in line with the random walk-based decision making in the PS approach. It is noteworthy that diffusion models have been successfully used in various cognitive tasks involving decision making (Shrager, Hogg, & Huberman, 1987; Ratcliff, Smith, Brown, & McKoon, 2016). Stella et al. (2019) show that hippocampal circuits can reactivate random trajectories of varying lengths and timescales that resemble Brownian diffusion. The NE process can also be interpreted as a kind of replay similar to the offline replay that contributes to generalization via multistep predictive representations of upcoming clips (or the successor representation; see Momennejad, Otto et al., 2017; Momennejad, Russek et al., 2017; Russek et al., 2017. It is different from online replay or multiple reflection in the PS model and closer to the offline replay that accommodates planning based on inferential piecing data together and multistep dependencies. The REMERGE (recurrency and episodic memory results in generalization) model of memory trace activation (Kumaran & McClelland, 2012) also uses replay and iterative updating of episodic memory for modeling rapid generalization in, for example, transitive inference task.

The final abilities of the E-EPS agent to master derived relations strongly depends on two parameters: α, which controls how much the NE affects the initially trained network, and β_t, which generates the probability distribution over the comparison stimuli. The post-processed network, $W_{t\to\infty}$, can be seen as an unconditioned network that a test trial can bias it. To account for the environment effect, we use a Bayesian approach and then apply the softmax function (see McClelland, 2013, for different models of contextual effects on perception). It is noteworthy that the PS model uses Bayesian updating, and therefore this update is in harmony with the PS agent (see Schwöbel, Kiebel, & Marković, 2018, and Parr, Markovic, Kiebel, & Friston, 2019, for modeling goal-directed behavior as an inference process).

The approach to the testing phase in the E-EPS model needs less computation at the decision time since it uses the cached updated network rather than processing the trained relations to compute derived relation links at each test trial.

In the rest of the letter, we discuss and conduct experiments on both models SNE and DNE, but the emphasis will be on the DNE, which we show is more effective than the SNE for the E-EPS model.

Table 1: Training Stages in Spencer and Chase (1996) Study: Number and Type of Training Trials.

	Number of Trials per Relation					
Training	*AB*	*BC*	*CD*	*DE*	*EF*	*FG*
AB	48					
BC	24	24				
CD	12	12	24			
DE	9	9	9	24		
EF	6	6	6	6	24	
FG	3	3	3	6	9	24
Baseline maintenance	3	3	3	3	3	3

4 Simulation Results

In this section, we study the model parameters in order to offer insights into how parameters can be tuned to simulate various behaviors, including typical human behavior or the behavior of people with some disabilities. To study the model in more detail, we consider a similar training setting as in the experiment by Spencer and Chase (1996), which Mofrad et al. (2020) address as well.

Spencer and Chase (1996) address the relatedness or nodal number using three seven-member stimulus classes. Stimuli are nonsense figures, and the training order is $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow G$. The training consists of seven stages as summarized in Table 1.[6] The first training block contains 48 trials of *AB* relations. Since there are three classes, the block for training, *AB*, contains 16 trials with the correct match A_1B_1, 16 trials with the correct match A_2B_2, and 16 trials with the correct match A_3B_3. The order of presented trials is random in the block, and the order of comparison stimuli, in this case B_1, B_2, B_3, is also randomly changed. If we consider the training of the *EF* relation, for instance, the training block contains six *AB* relations (which means each trial with A_1B_1, A_2B_2, and A_3B_3 as the correct pair appears twice), 6 *BC* relations (each trial with B_1C_1, B_2C_2, and B_3C_3 as the correct pair appears twice), 6 *CD* relations and 6 *DE* relations, and finally the new relation *EF* with 24 relations (i.e., each trial with E_1F_1, E_2F_2, and E_3F_3 as the correct pair appears eight times). In the baseline maintenance stage, no new relation is provided and each correct relation appears only once. The

[6] It is noteworthy that in Spencer and Chase (1996), each stage of training has 48 trials. To ease the simulation, the fourth stage for *DE* training is changed, so we consider 9 instead of 8 trials for *AB*, *BC*, and *CD* relations. Therefore, this stage has 51 trials in the simulation instead of the original 48.

mastery criterion is set to 0.9, and if the agent passes the mastery criterion for all stages and the final baseline maintenance, then we can test the agent for formation of derived relations.

The reported simulation results are the average over 1000 simulations.

4.1 Experiment 1: Step-by-Step Process. In this experiment, we illustrate the computation steps. In Figure 1a, the network h-values after the training phase (based on Table 1) is depicted where the parameters are set to $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0.7$. Note that the symmetry and reflexivity connections in addition to the baseline connections appear in Figure 1a. The reflexivity h-values are the maximum h-value at each row (input-output connections). Moreover, since $K = 1$, the W_h matrix is symmetric—for instance, $A_1B_1 = B_1A_1 = 51.82$. To compute the transition probability matrix, a softmax function with parameter $\beta_h = 0.1$ is used. Note that the transition probability matrix is just row-normalized and not symmetric. All the reported values are rounded by two or three decimal places.

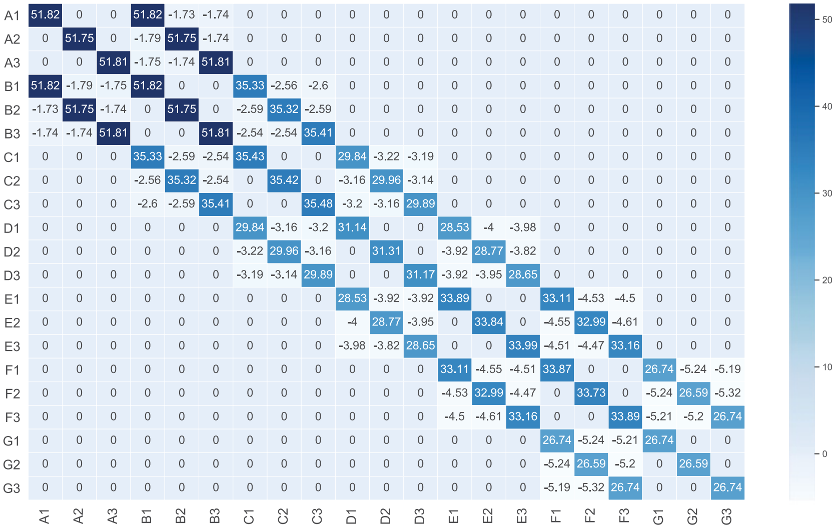
We set $W_0 = P$ as the input matrix to the NE. We might use P (Figure 1b) for the iterative updates (DNE) or $\mathcal{T}$ matrix (SNE). In Figure 2, we address DNE when $\alpha = 0.7$. The convergence criterion is that $\sum_{i,j} |W_{t+1} - W_t|_{i,j} < 0.0001$. One can also compute the converged network $W_{t\to\infty}$ using the theoretical converged formula provided in equation 3.3.

Figure 2a shows the general internal map of the network clip before the testing phase. One can interpret these values as how the stimuli are prioritized in the agent memory when there is no external trial that measures the accuracy of answers in MTS trials. Figure 2b shows the performance of the agent when it comes to the testing phase. For instance, if the sample stimulus is A_3 and the comparison stimuli are F_1, F_2, and F_3, then the agent chooses F_1 and F_2 with probability 0.092 and selects F_3 with probability 0.815.

To calculate these category-based probability distributions, first the conditional probability for any specific category is calculated based on Bayes' rule, and then the softmax function transfers these vectors to the desired probabilities based on the chosen parameter β_t. The conditional input can show the context, or environment, effect, and therefore we can apply the same β_t as a characteristic of the agent for all the categories.

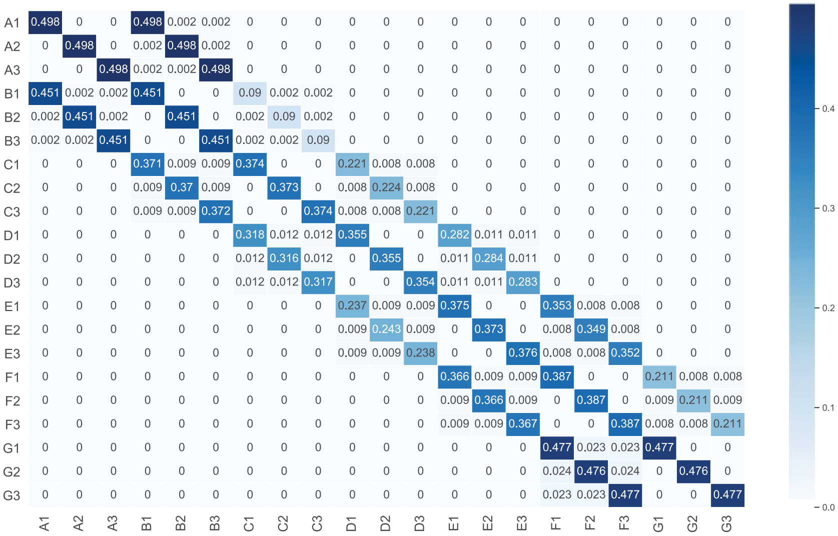
If we use SNE, first we have to compute $\mathcal{T}$, which is reported in Figure 3a and then update the network using $\alpha = 0.7$ parameter. The localized network $\mathcal{T}$ adds weights to the one-node relations, and we have two more diagonals in $\mathcal{T}$ in comparison with P.

The goal of this experiment is to illustrate how both DNE and SNE are working. In experiment 2, we compare the two updating methods for symmetry and transitivity relations and discuss why DNE could be a more appropriate option for enhancing the EPS model.



	A1	A2	A3	B1	B2	B3	C1	C2	C3	D1	D2	D3	E1	E2	E3	F1	F2	F3	G1	G2	G3
A1	51.82	0	0	51.82	-1.73	-1.74	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A2	0	51.75	0	-1.79	51.75	-1.74	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A3	0	0	51.81	-1.75	-1.74	51.81	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B1	51.82	-1.79	-1.75	51.82	0	0	35.33	-2.56	-2.6	0	0	0	0	0	0	0	0	0	0	0	0
B2	-1.73	51.75	-1.74	0	51.75	0	-2.59	35.32	-2.59	0	0	0	0	0	0	0	0	0	0	0	0
B3	-1.74	-1.74	51.81	0	0	51.81	-2.54	-2.54	35.41	0	0	0	0	0	0	0	0	0	0	0	0
C1	0	0	0	35.33	-2.59	-2.54	35.43	0	0	29.84	-3.22	-3.19	0	0	0	0	0	0	0	0	0
C2	0	0	0	-2.56	35.32	-2.54	0	35.42	0	-3.16	29.96	-3.14	0	0	0	0	0	0	0	0	0
C3	0	0	0	-2.6	-2.59	35.41	0	0	35.48	-3.2	-3.16	29.89	0	0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	29.84	-3.16	-3.2	31.14	0	0	28.53	-4	-3.98	0	0	0	0	0	0
D2	0	0	0	0	0	0	-3.22	29.96	-3.16	0	31.31	0	-3.92	28.77	-3.82	0	0	0	0	0	0
D3	0	0	0	0	0	0	-3.19	-3.14	29.89	0	0	31.17	-3.92	-3.95	28.65	0	0	0	0	0	0
E1	0	0	0	0	0	0	0	0	0	28.53	-3.92	-3.92	33.89	0	0	33.11	-4.53	-4.5	0	0	0
E2	0	0	0	0	0	0	0	0	0	-4	28.77	-3.95	0	33.84	0	-4.55	32.99	-4.61	0	0	0
E3	0	0	0	0	0	0	0	0	0	-3.98	-3.82	28.65	0	0	33.99	-4.51	-4.47	33.16	0	0	0
F1	0	0	0	0	0	0	0	0	0	0	0	0	33.11	-4.55	-4.51	33.87	0	0	26.74	-5.24	-5.19
F2	0	0	0	0	0	0	0	0	0	0	0	0	-4.53	32.99	-4.47	0	33.73	0	-5.24	26.59	-5.32
F3	0	0	0	0	0	0	0	0	0	0	0	0	-4.5	-4.61	33.16	0	0	33.89	-5.21	-5.2	26.74
G1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	26.74	-5.24	-5.21	26.74	0	0
G2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-5.24	26.59	-5.2	0	26.59	0
G3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	-5.19	-5.32	26.74	0	0	26.74

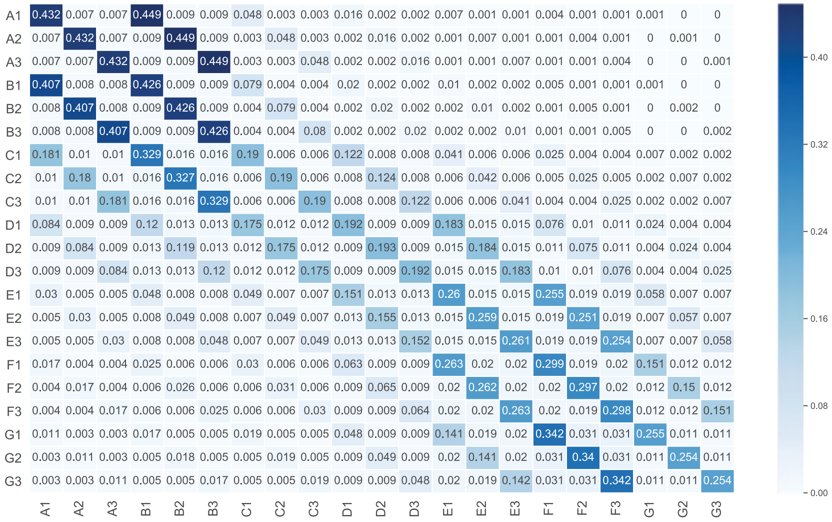
(a) Network clip W_h, composed of h-values at the end of training phase.



	A1	A2	A3	B1	B2	B3	C1	C2	C3	D1	D2	D3	E1	E2	E3	F1	F2	F3	G1	G2	G3
A1	0.498	0	0	0.498	0.002	0.002	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A2	0	0.498	0	0.002	0.498	0.002	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A3	0	0	0.498	0.002	0.002	0.498	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B1	0.451	0.002	0.002	0.451	0	0	0.09	0.002	0.002	0	0	0	0	0	0	0	0	0	0	0	0
B2	0.002	0.451	0.002	0	0.451	0	0.002	0.09	0.002	0	0	0	0	0	0	0	0	0	0	0	0
B3	0.002	0.002	0.451	0	0	0.451	0.002	0.002	0.09	0	0	0	0	0	0	0	0	0	0	0	0
C1	0	0	0	0.371	0.009	0.009	0.374	0	0	0.221	0.008	0.008	0	0	0	0	0	0	0	0	0
C2	0	0	0	0.009	0.37	0.009	0	0.373	0	0.008	0.224	0.008	0	0	0	0	0	0	0	0	0
C3	0	0	0	0.009	0.009	0.372	0	0	0.374	0.008	0.008	0.221	0	0	0	0	0	0	0	0	0
D1	0	0	0	0	0	0	0.318	0.012	0.012	0.355	0	0	0.282	0.011	0.011	0	0	0	0	0	0
D2	0	0	0	0	0	0	0.012	0.316	0.012	0	0.355	0	0.011	0.284	0.011	0	0	0	0	0	0
D3	0	0	0	0	0	0	0.012	0.012	0.317	0	0	0.354	0.011	0.011	0.283	0	0	0	0	0	0
E1	0	0	0	0	0	0	0	0	0	0.237	0.009	0.009	0.375	0	0	0.353	0.008	0.008	0	0	0
E2	0	0	0	0	0	0	0	0	0	0.009	0.243	0.009	0	0.373	0	0.008	0.349	0.008	0	0	0
E3	0	0	0	0	0	0	0	0	0	0.009	0.009	0.238	0	0	0.376	0.008	0.008	0.352	0	0	0
F1	0	0	0	0	0	0	0	0	0	0	0	0	0.366	0.009	0.009	0.387	0	0	0.211	0.008	0.008
F2	0	0	0	0	0	0	0	0	0	0	0	0	0.009	0.366	0.009	0	0.387	0	0.009	0.211	0.009
F3	0	0	0	0	0	0	0	0	0	0	0	0	0.009	0.009	0.367	0	0	0.387	0.008	0.008	0.211
G1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.477	0.023	0.023	0.477	0	0
G2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.024	0.476	0.024	0	0.476	0
G3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.023	0.023	0.477	0	0	0.477

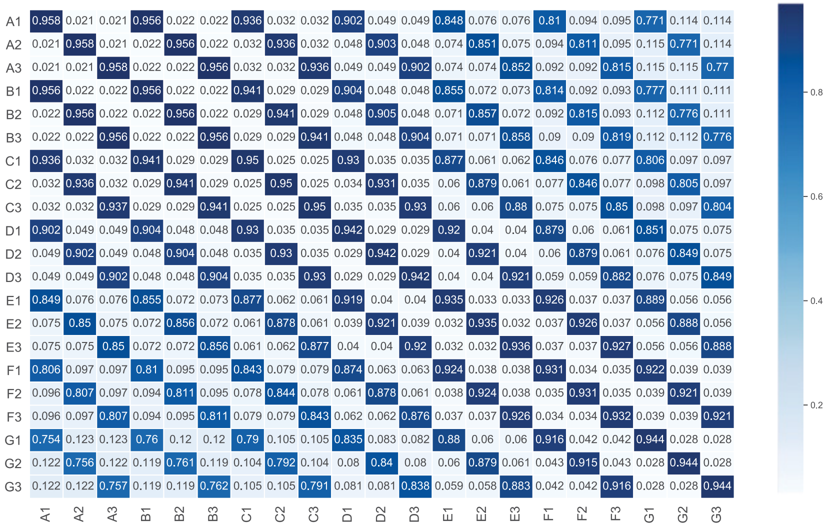
(b) The transition probability matrix P using $\beta_h = 0.1$. The reported values are rounded by three decimal places.

Figure 1: A sample configuration of network h-values after training $A \to B \to C \to D \to E \to F \to G$ when $\gamma = 0.001$, $K = 1$, and $\beta_h = 0.1$.



	A1	A2	A3	B1	B2	B3	C1	C2	C3	D1	D2	D3	E1	E2	E3	F1	F2	F3	G1	G2	G3
A1	0.432	0.007	0.007	0.449	0.009	0.009	0.048	0.003	0.003	0.016	0.002	0.002	0.007	0.001	0.001	0.004	0.001	0.001	0.001	0	0
A2	0.007	0.432	0.007	0.009	0.449	0.009	0.003	0.048	0.003	0.002	0.016	0.002	0.001	0.007	0.001	0.001	0.004	0.001	0	0.001	0
A3	0.007	0.007	0.432	0.009	0.009	0.449	0.003	0.003	0.048	0.002	0.002	0.016	0.001	0.001	0.007	0.001	0.001	0.004	0	0	0.001
B1	0.407	0.008	0.008	0.426	0.009	0.009	0.079	0.004	0.004	0.02	0.002	0.002	0.01	0.002	0.002	0.005	0.001	0.001	0.001	0	0
B2	0.008	0.407	0.008	0.009	0.426	0.009	0.004	0.079	0.004	0.002	0.02	0.002	0.002	0.01	0.002	0.001	0.005	0.001	0	0.002	0
B3	0.008	0.008	0.407	0.009	0.009	0.426	0.004	0.004	0.08	0.002	0.002	0.02	0.002	0.002	0.01	0.001	0.001	0.005	0	0	0.002
C1	0.181	0.01	0.01	0.329	0.016	0.016	0.19	0.006	0.006	0.122	0.008	0.008	0.041	0.006	0.006	0.025	0.004	0.004	0.007	0.002	0.002
C2	0.01	0.18	0.01	0.016	0.327	0.016	0.006	0.19	0.006	0.008	0.124	0.008	0.006	0.042	0.006	0.005	0.025	0.005	0.002	0.007	0.002
C3	0.01	0.01	0.181	0.016	0.016	0.329	0.006	0.006	0.19	0.008	0.008	0.122	0.006	0.006	0.041	0.004	0.004	0.025	0.002	0.002	0.007
D1	0.084	0.009	0.009	0.12	0.013	0.013	0.175	0.012	0.012	0.192	0.009	0.009	0.183	0.015	0.015	0.076	0.01	0.011	0.024	0.004	0.004
D2	0.009	0.084	0.009	0.013	0.119	0.013	0.012	0.175	0.012	0.009	0.193	0.009	0.015	0.184	0.015	0.011	0.075	0.011	0.004	0.024	0.004
D3	0.009	0.009	0.084	0.013	0.013	0.12	0.012	0.012	0.175	0.009	0.009	0.192	0.015	0.015	0.183	0.01	0.01	0.076	0.004	0.004	0.025
E1	0.03	0.005	0.005	0.048	0.008	0.008	0.049	0.007	0.007	0.151	0.013	0.013	0.26	0.015	0.015	0.255	0.019	0.019	0.058	0.007	0.007
E2	0.005	0.03	0.005	0.008	0.049	0.008	0.007	0.049	0.007	0.013	0.155	0.013	0.015	0.259	0.015	0.019	0.251	0.019	0.007	0.057	0.007
E3	0.005	0.005	0.03	0.008	0.008	0.048	0.007	0.007	0.049	0.013	0.013	0.152	0.015	0.015	0.261	0.019	0.019	0.254	0.007	0.007	0.058
F1	0.017	0.004	0.004	0.025	0.006	0.006	0.03	0.006	0.006	0.063	0.009	0.009	0.263	0.02	0.02	0.299	0.019	0.02	0.151	0.012	0.012
F2	0.004	0.017	0.004	0.006	0.026	0.006	0.006	0.031	0.006	0.009	0.065	0.009	0.02	0.262	0.02	0.02	0.297	0.02	0.012	0.15	0.012
F3	0.004	0.004	0.017	0.006	0.006	0.025	0.006	0.006	0.03	0.009	0.009	0.064	0.02	0.02	0.263	0.02	0.019	0.298	0.012	0.012	0.151
G1	0.011	0.003	0.003	0.017	0.005	0.005	0.019	0.005	0.005	0.048	0.009	0.009	0.141	0.019	0.02	0.342	0.031	0.031	0.255	0.011	0.011
G2	0.003	0.011	0.003	0.005	0.018	0.005	0.005	0.019	0.005	0.009	0.049	0.009	0.02	0.141	0.02	0.031	0.34	0.031	0.011	0.254	0.011
G3	0.003	0.003	0.011	0.005	0.005	0.017	0.005	0.005	0.019	0.009	0.009	0.048	0.02	0.019	0.142	0.031	0.031	0.342	0.011	0.011	0.254

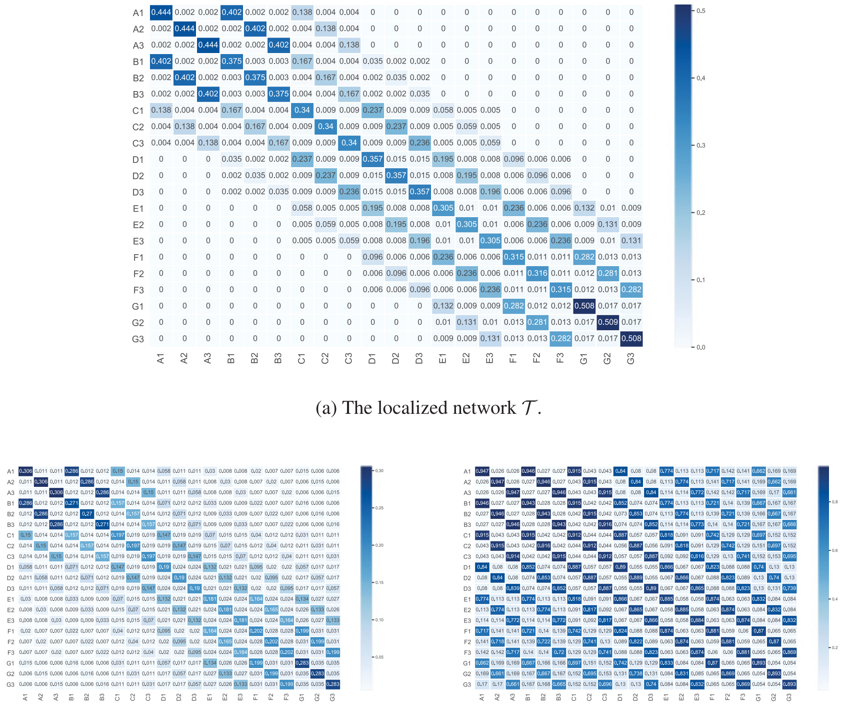
(a) Converged network $W_{t\to\infty}$ using $\alpha = 0.7$.



	A1	A2	A3	B1	B2	B3	C1	C2	C3	D1	D2	D3	E1	E2	E3	F1	F2	F3	G1	G2	G3
A1	0.958	0.021	0.021	0.956	0.022	0.022	0.936	0.032	0.032	0.902	0.049	0.049	0.848	0.076	0.076	0.81	0.094	0.095	0.771	0.114	0.114
A2	0.021	0.958	0.021	0.022	0.956	0.022	0.032	0.936	0.032	0.048	0.903	0.048	0.074	0.851	0.075	0.094	0.811	0.095	0.115	0.771	0.114
A3	0.021	0.021	0.958	0.022	0.022	0.956	0.032	0.032	0.936	0.049	0.049	0.902	0.074	0.074	0.852	0.092	0.092	0.815	0.115	0.115	0.77
B1	0.956	0.022	0.022	0.956	0.022	0.022	0.941	0.029	0.029	0.904	0.048	0.048	0.855	0.072	0.073	0.814	0.092	0.093	0.777	0.111	0.111
B2	0.022	0.956	0.022	0.022	0.956	0.022	0.029	0.941	0.029	0.048	0.905	0.048	0.071	0.857	0.072	0.092	0.815	0.093	0.112	0.776	0.111
B3	0.022	0.022	0.956	0.022	0.022	0.956	0.029	0.029	0.941	0.048	0.048	0.904	0.071	0.071	0.858	0.09	0.09	0.819	0.112	0.112	0.776
C1	0.936	0.032	0.032	0.941	0.029	0.029	0.95	0.025	0.025	0.93	0.035	0.035	0.877	0.061	0.062	0.846	0.076	0.077	0.806	0.097	0.097
C2	0.032	0.936	0.032	0.029	0.941	0.029	0.025	0.95	0.025	0.034	0.931	0.035	0.06	0.879	0.061	0.077	0.846	0.077	0.098	0.805	0.097
C3	0.032	0.032	0.937	0.029	0.029	0.941	0.025	0.025	0.95	0.035	0.035	0.93	0.06	0.06	0.88	0.075	0.075	0.85	0.098	0.097	0.804
D1	0.902	0.049	0.049	0.904	0.048	0.048	0.93	0.035	0.035	0.942	0.029	0.029	0.92	0.04	0.04	0.879	0.06	0.061	0.851	0.075	0.075
D2	0.049	0.902	0.049	0.048	0.904	0.048	0.035	0.93	0.035	0.029	0.942	0.029	0.04	0.921	0.04	0.06	0.879	0.061	0.076	0.849	0.075
D3	0.049	0.049	0.902	0.048	0.048	0.904	0.035	0.035	0.93	0.029	0.029	0.942	0.04	0.04	0.921	0.059	0.059	0.882	0.076	0.075	0.849
E1	0.849	0.076	0.076	0.855	0.072	0.073	0.877	0.062	0.061	0.919	0.04	0.04	0.935	0.033	0.033	0.926	0.037	0.037	0.889	0.056	0.056
E2	0.075	0.85	0.075	0.072	0.856	0.072	0.061	0.878	0.061	0.039	0.921	0.039	0.032	0.935	0.032	0.037	0.926	0.037	0.056	0.888	0.056
E3	0.075	0.075	0.85	0.072	0.072	0.856	0.061	0.062	0.877	0.04	0.04	0.92	0.032	0.032	0.936	0.037	0.037	0.927	0.056	0.056	0.888
F1	0.806	0.097	0.097	0.81	0.095	0.095	0.843	0.079	0.079	0.874	0.063	0.063	0.924	0.038	0.038	0.931	0.034	0.035	0.922	0.039	0.039
F2	0.096	0.807	0.097	0.094	0.811	0.095	0.078	0.844	0.078	0.061	0.878	0.061	0.038	0.924	0.038	0.035	0.931	0.035	0.039	0.921	0.039
F3	0.096	0.097	0.807	0.094	0.095	0.811	0.079	0.079	0.843	0.062	0.062	0.876	0.037	0.037	0.926	0.034	0.034	0.932	0.039	0.039	0.921
G1	0.754	0.123	0.123	0.76	0.12	0.12	0.79	0.105	0.105	0.835	0.083	0.082	0.88	0.06	0.06	0.916	0.042	0.042	0.944	0.028	0.028
G2	0.122	0.756	0.122	0.119	0.761	0.119	0.104	0.792	0.104	0.08	0.84	0.08	0.06	0.879	0.061	0.043	0.915	0.043	0.028	0.944	0.028
G3	0.122	0.122	0.757	0.119	0.119	0.762	0.105	0.105	0.791	0.081	0.081	0.838	0.059	0.058	0.883	0.042	0.042	0.916	0.028	0.028	0.944

(b) Category-based probability distributions for the test phase using $\beta_t = 4$.

Figure 2: The new network adjacency matrix when the regularization parameter is $\alpha = 0.7$ with the input matrix $W_0 = P$, which is given in Figure 1b. The test phase probabilities in Figure 2b are calculated by normalizing the weights for the specific category and then using the softmax function with parameter $\beta_t = 4$.



(a) The localized network $\mathcal{T}$.

(b) Converged network $W_{t\to\infty}$ using $\alpha = 0.7$.

(c) Category-based probability distributions for the test phase using $\beta_t = 4$.

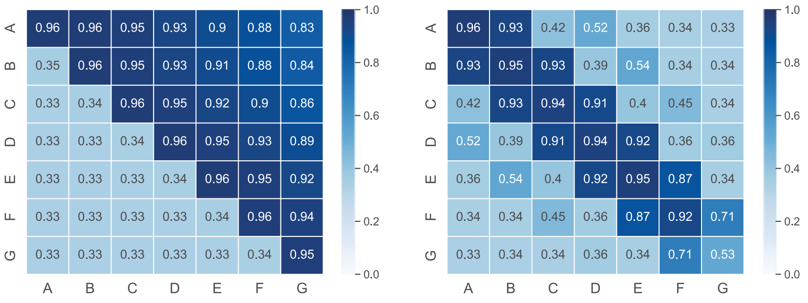
Figure 3: The new network adjacency matrix using an SNE update when the regularization parameter is $\alpha = 0.7$ and the input matrix $W_0 = P$, which is given in Figure 1b. The test phase probabilities in Figure 3c are calculated by normalizing the weights for the specific category and then using the softmax function with parameter $\beta_t = 4$.

4.2 Experiment 2: Isolating Symmetry and Transitivity.

Two main differences between DNE and SNE are shown in this experiment. In this regard, we consider two extreme cases to isolate the symmetry and transitivity effects.

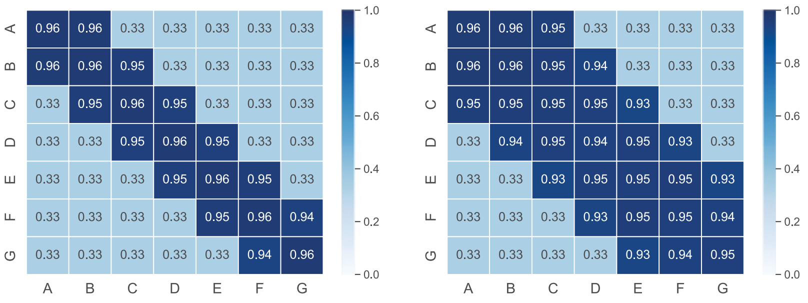
First, we isolate the effect of symmetry relations; in other words, we suppose that the agent is able to answer the transitive relations but unable to derive symmetry relations. For this, we set the parameters to $\gamma = 0.001$, $K = 0.01$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0.05$.

As illustrated in Figure 4a, the symmetry relations, and therefore the equivalence relations, can be altered by parameter K. However, in Figure 4b, due to the symmetric behavior of updates, the symmetry relations are exactly the same as the baseline relations, and the transitive and equivalence



(a) Final category-based results applying DNE (b) Final category-based results applying SNE

Figure 4: The probability of choosing correct pairs between categories when $\gamma = 0.001$, $K = 0.01$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0.05$. The reported values are calculated by taking the average over all relations in each category.



(a) Final category-based results applying DNE (b) Final category-based results applying SNE

Figure 5: The probability of choosing correct pairs between categories when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0$.

relations are altered by setting $K = 0.01$. We can conclude that a DNE-type agent can handle nonsymmetric relations, but the SNE agent is unable to control symmetry relations independently.

Next, we simulate a scenario in which the agent learns the baseline relations, but no transitive relation is derived. Suppose the symmetry relations are derived perfectly, so that we only isolate the transitive relations. Let the parameters of such an agent be $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0$.

In Figure 5a, which uses the DNE method, the transitive and therefore equivalence relations are not formed, while the symmetry relations are strong. In Figure 5b, we see that the one-node relations such as *AC* and *BD* are derived in SNE. This is expected due to the definition of $\mathcal{T}$. In the EPS model, though, we are seeking to control all the transitive and equivalence relations.

Table 2: The Average of Required Repetition of Training Blocks until Reaching Mastery Criterion Ratio 0.9 When $\gamma = 0.001$, $K = 1$, $\beta_t = 4$, and $\alpha = 0.05$ for Three Values of $\beta_h = 0.2, 0.1,$ and 0.05.

	Number of Trials per Relation						Time		
Training	*AB*	*BC*	*CD*	*DE*	*EF*	*FG*	$\beta_h = 0.2$	$\beta_h = 0.1$	$\beta_h = 0.05$
AB	48						2.133	3.423	5.907
BC	24	24					2.885	4.757	8.751
CD	12	12	24				2.959	4.977	9.641
DE	9	9	9	24			2.791	4.661	9.469
EF	6	6	6	6	24		2.992	5.208	11.736
FG	3	3	3	6	9	24	3.008	5.339	12.978
Baseline maintenance	3	3	3	3	3	3	1.038	1.407	7.561

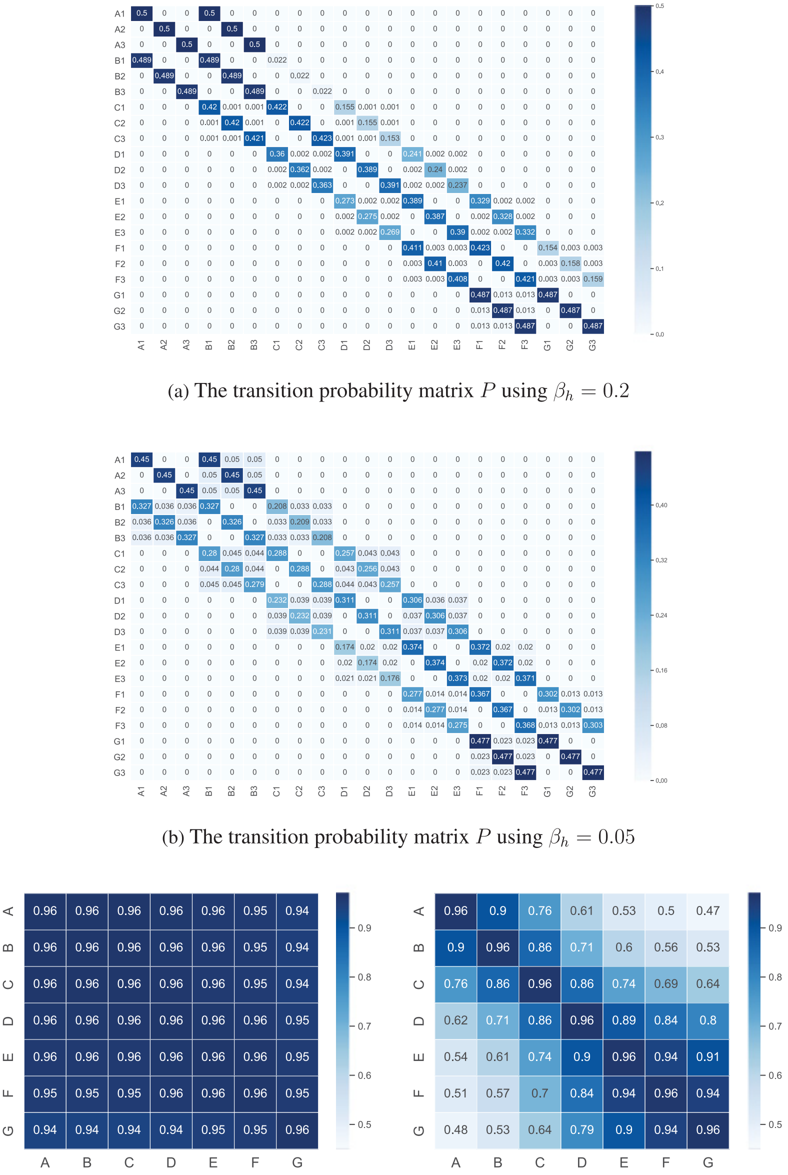
Therefore, since SNE is not an appropriate method for controlling symmetry and transitivity completely, we consider DNE as the main approach in this letter to cover more general cases, such as those with weak symmetry relations or weak transitivity relations. In the rest of the simulations, we report just the results for the DNE method.

4.3 Experiment 3: Effect of the β_h Parameter. The softmax function parameter β_h is used in the training phase for checking the mastery criterion as well as computing the transition matrix from W_h. As reported in Table 2, a higher value of β_h causes the agent to be able to pass the training phase faster, while for smaller values of β_h, it takes many more iterations to pass the training phase and learn baseline relations. Table 2 presents the learning speed for three values of $\beta_h = 0.2, 0.1,$ and 0.05 when $\gamma = 0.001$, $K = 1$, $\beta_t = 4$, and $\alpha = 0.05$.

Table 2 shows that parameter β_h can be used to control the learning speed. For instance, an agent with $\beta_h = 0.2$ learns *AB* relations by repeating the training blocks 2.1 times on average. This value will be 3.4 for $\beta_h = 0.1$ and 5.9 for $\beta_h = 0.05$.

Another effect of β_h appears in computing the probability matrix and, consequently, the final network shape. In Figure 6, we report the *P* matrix and the computed nodal effect in the test phase for two choices of $\beta_h = 0.2$ and $\beta_h = 0.05$ when we keep all parameters similar: $\gamma = 0.001$, $K = 1$, $\beta_t = 4$, and $\alpha = 0.05$.

By comparing Figures 6a and 6b, we notice that the probability of direct relations are weaker when $\beta_h = 0.05$. Since this matrix is considered as W_0, the input matrix to the NE iterative method, the final results will be altered. In Figure 6c, the nodal effect is negligible, and all the transitive and equivalence relations are formed equally well as baseline relations. Figure 6d, however, shows the nodal effect and the agent's weak performance in



(a) The transition probability matrix P using $\beta_h = 0.2$

(b) The transition probability matrix P using $\beta_h = 0.05$

(c) Final category-based probability of correct choice in the test phase when $\beta_h = 0.2$

(d) Final category-based probability of correct choice in the test phase when $\beta_h = 0.05$

Figure 6: Comparison of probability matrix out of training and final category-based probability of correct choice in the test phase for two choices of $\beta_h = 0.2$ and $\beta_h = 0.05$, when $\gamma = 0.001$, $K = 1$, $\beta_t = 4$, and $\alpha = 0.05$.

Table 3: The Average of Required Repetition of Training Blocks until Reaching Mastery Criterion Ratio 0.9 When $K = 1$, $\beta_h = 0.1$, $\beta_t = 4$, and $\alpha = 0.05$ for Three Values of $\gamma = 0, 0.001,$ and 0.005.

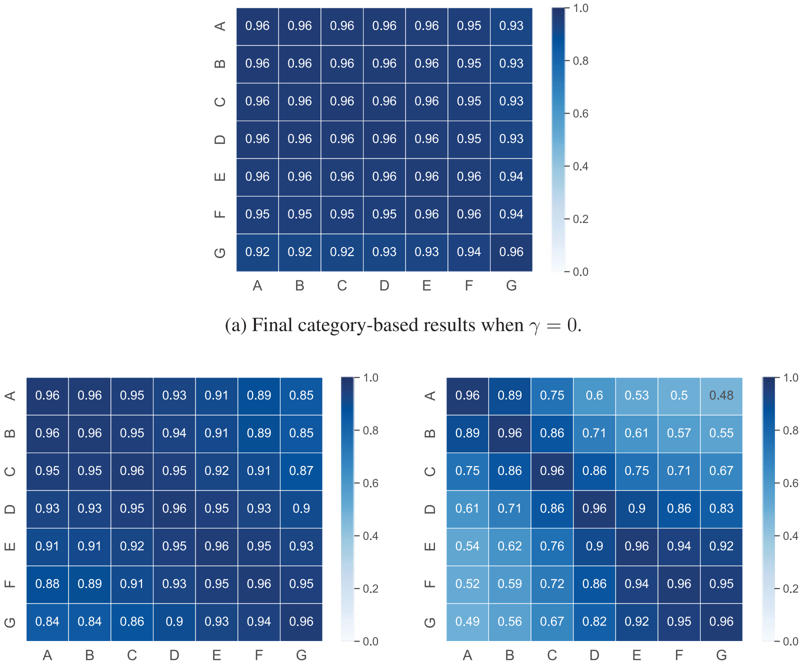
	Number of Trials per Relation						Time		
Training	*AB*	*BC*	*CD*	*DE*	*EF*	*FG*	$\gamma = 0.0$	$\gamma = 0.001$	$\gamma = 0.002$
AB	48						3.318	3.452	3.580
BC	24	24					4.391	4.703	5.088
CD	12	12	24				4.570	4.951	5.584
DE	9	9	9	24			4.200	4.654	5.514
EF	6	6	6	6	24		4.649	5.190	6.951
FG	3	3	3	6	9	24	4.637	5.324	7.884
Baseline maintenance	3	3	3	3	3	3	1.089	1.414	7.281

relations with a higher nodal number. We conclude that β_h can be used for controlling both the speed of learning and the nodal effect. In other words, if we fix all other parameters than β_h, the smaller value of β_h results in slower learning and a lower chance of forming transitive and equivalence relations with a higher nodal number. It is noteworthy that the effects of β_h and γ are somehow intertwined. As we see in experiment 4, γ also controls the learning speed and nodal effect. Indeed, if the agent does not forget at all, that is, $\gamma = 0$, then β_h controls just the speed of learning. However, $\gamma = 0$ is not a plausible choice for replication of human behavior.

4.4 Experiment 4: Effect of the γ Parameter. Mofrad et al. (2020) studied, the effect of γ in the training phase of EPS agents, where learning speed can be adjusted via γ. In Table 3, the average number of repetitions at each stage is provided for three choices: $\gamma = 0, 0.001,$ and 0.005. There is a general trend that increasing the forgetting factor will increase the repetition times in all stages. But the rates of increase for later stages and the baseline maintenance are different. The explanation is that the forgetting factor affects the initial learned relations more since at the final blocks, we have fewer of them. In other words, in the final blocks, we have fewer trials of them, and thus the forgetting factor will cause a stronger adverse impact. This is why we need around seven iterations of the maintenance phase when $\gamma = 0.002$, while we need just one iteration by removing the forgetting factor, $\gamma = 0$.

The forgetting factor will affect the final shape of h-values network W_h, and therefore for similar parameters, we have different probability matrices and final outcomes in the test phase. Figure 7 provides the final results of the testing phase for three different values of the forgetting factor: $\gamma = 0, 0.001, 0.002$.

When $\gamma = 0$ (see Figure 7a), there is no forgetting, and therefore the training order does not matter and all the relations are considered equally the



(a) Final category-based results when $\gamma = 0$.

(b) Final category-based results when $\gamma = 0.001$

(c) Final category-based results when $\gamma = 0.002$

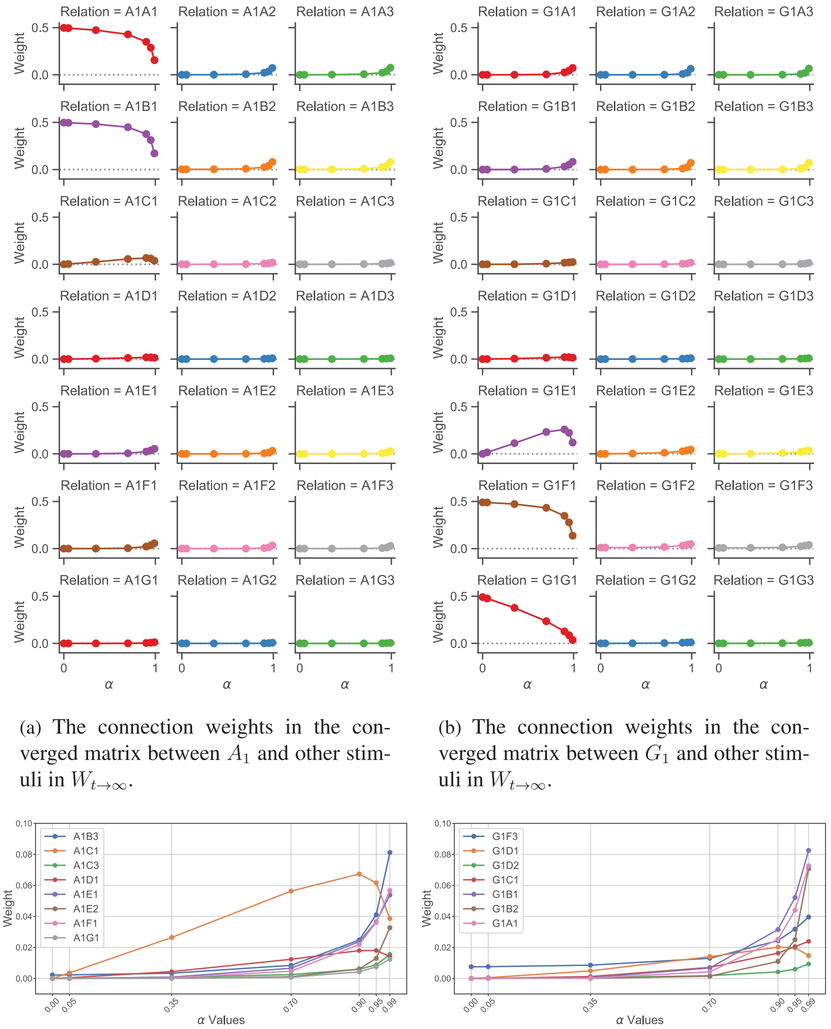
Figure 7: Probability of choosing correct pairs between categories when $K = 1$, $\beta_h = 0.1, \beta_t = 4$, and $\alpha = 0.05$ for three forgetting factor values: $\gamma = 0, 0.001$, and 0.002.

same. In Figure 7b, all the relations are formed but we can easily notice the nodal effect. For instance, if we test the *AB* relation, the probability of a correct choice by the agent is 0.96, while it is about 0.85 for *AG* with five nodes in between. Figure 7c shows that a higher forgetting factor can be used to model impaired equivalence class formation. If we test the agent with the *AB* relation, the probability of a correct choice would be 0.89, while it is about 0.48 for *AG*. Comparing the correct choice probabilities for *AB* and *FG* (0.89 for *AB* versus 0.95 for *FG*) shows the importance of training order in this setting. The agent forgets the initial stage relations, and these relations need to be repeated. If the training trial blocks are totally separate, as in experiment 1 in Mofrad et al. (2020), the initial trained relations drop dramatically with a high forgetting factor.

To show the importance of testing order in the model, similar to the SE literature, we simulate the testing phase with different test orders so the trials that appear late in the testing phase have weaker results when the forgetting factor is high. Here, for simplicity, we calculate the probability distribution for different test trials and evaluate the agent behavior based on them. This means the forgetting factor is not effective on the test results in the current simulations. However, the forgetting factor can be used by defining β_t as a function of time and γ to model the forgetting in the testing phase of E-EPS. Another argument is that the forgetting might affect the network; in this case, the network weights must be updated in a way to keep each row summing to one. Therefore, it is not as straightforward as the EPS where the matrix with h-values is the basis for the testing phase.

4.5 Experiment 5: Effect of the α Parameter. This parameter shapes the final representation of the clip network (see appendix A for a theoretical discussion). A smaller value of α biases the converged matrix $W_{t \to \infty}$ to keep the connections from W_0 stronger, while a larger value of α enhances transitive relations. In the case of $\alpha = 0$, as represented in Figure 5a, there is no enhancement in the network using DNE. Figures 8a and 8b, respectively, represent the connection values from A_1 and G_1 to other stimuli in the converged network for $\alpha = 0, 0.05, 0.35, 0.7, 0.9, 0.95,$ and 0.99, when $\gamma = 0.001$, $K = 1$, and $\beta_h = 0.1$.

As depicted in Figure 8, smaller values of α keep the relations in the input network (i.e., trained relations together with symmetry and reflexivity) stronger. On the other hand, a higher α value reinforces the transitive and equivalence relations. For each α value, the connection weights for all relations must sum to one; for instance, the values for $\alpha = 0.9$ in all subplots of Figure 8a sum to one as they show the transition probability from A_1 to all other points when using $\alpha = 0.9$. As a result, increasing the values for transitive relations means a decrease in initial relations (see the decrease in A_1A_1, A_1B_1 relation weights and the increase in other values, say, A_1C_1 and A_1G_1). Along with construction and enhancing the desired relations (see the first columns in Figures 8a and 8b), the undesired relations are also constructed and enhanced to some extent. This can be explained by the fact that the values for undesired relations such as $A_1B_2, A_1B_3, G_1F_2,$ and G_1F_3 are not zero in the initial matrix since the training criterion was set to 0.9. These values could enhance undesired relations, especially when α is higher. For instance, as depicted in Figure 8c, the connection weight for the A_1C_1 relation, which is a desired relation, decreases for α values higher than 0.9. Similarly, the connection weight for the A_1D_1 relation decreases at $\alpha = 0.99$ in comparison with $\alpha = 0.9, 0.95$. The connection weight for the A_1B_3 relation, which has a very small weight in the beginning (i.e., when $\alpha = 0$), increases with α with acceleration in the rate of change for α values greater than 0.7. A_1C_3 and A_1E_2 are two sample relations that are undesirable and get enhanced during the diffusion process as a function of α value. The same kind of behavior can be observed for relations from G_1. In Figure 8d, the relation G_1D_1 increases as desired, but when α is too high ($\alpha = 0.95, 0.99$), it starts to decrease. Undesired relations such as G_1F_3 and G_1D_2 are enhanced with



(a) The connection weights in the converged matrix between A_1 and other stimuli in $W_{t\to\infty}$.

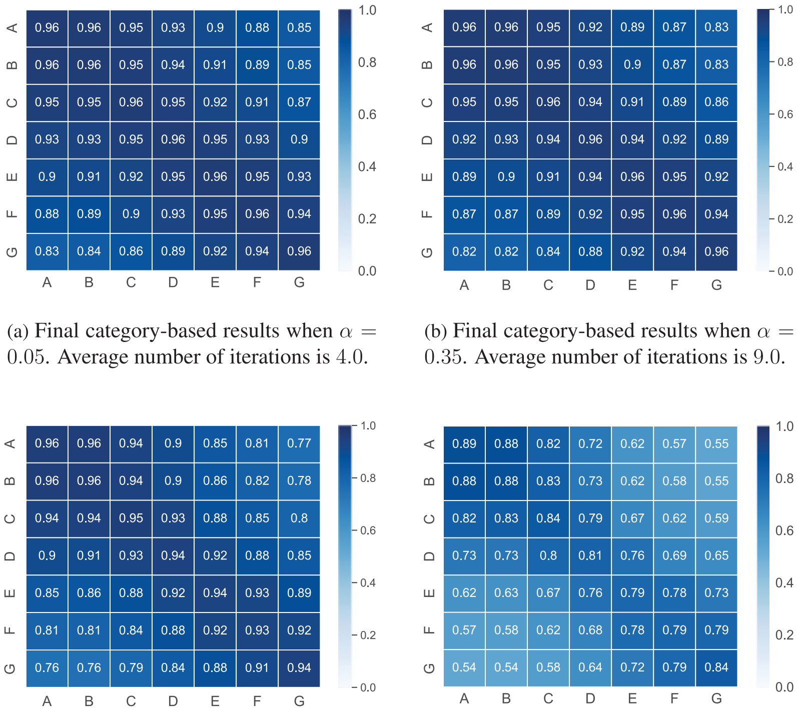
(b) The connection weights in the converged matrix between G_1 and other stimuli in $W_{t\to\infty}$.

(c) A comparison between behavior of some desired and undesired relations between A_1 and other stimuli based on different α values.

(d) A comparison between behavior of some desired and undesired relations between G_1 and other stimuli based on different α values.

Figure 8: The connection weights in the converged matrix $W_{t\to\infty}$ for A_1 and G_1 for $\alpha = 0, 0.05, 0.35, 0.7, 0.9, 0.95, 0.99$, when $\gamma = 0.001$, $K = 1$, and $\beta_h = 0.1$.

a higher rate when α approaches one. Therefore, an inappropriate choice of α could be destructive; in this example, a higher value of α than 0.9 sounds inappropriate.



	A	B	C	D	E	F	G
A	0.96	0.96	0.95	0.93	0.9	0.88	0.85
B	0.96	0.96	0.95	0.94	0.91	0.89	0.85
C	0.95	0.95	0.96	0.95	0.92	0.91	0.87
D	0.93	0.93	0.95	0.96	0.95	0.93	0.9
E	0.9	0.91	0.92	0.95	0.96	0.95	0.93
F	0.88	0.89	0.9	0.93	0.95	0.96	0.94
G	0.83	0.84	0.86	0.89	0.92	0.94	0.96

(a) Final category-based results when $\alpha = 0.05$. Average number of iterations is 4.0.

	A	B	C	D	E	F	G
A	0.96	0.96	0.95	0.92	0.89	0.87	0.83
B	0.96	0.96	0.95	0.93	0.9	0.87	0.83
C	0.95	0.95	0.96	0.94	0.91	0.89	0.86
D	0.92	0.93	0.94	0.96	0.94	0.92	0.89
E	0.89	0.9	0.91	0.94	0.96	0.95	0.92
F	0.87	0.87	0.89	0.92	0.95	0.96	0.94
G	0.82	0.82	0.84	0.88	0.92	0.94	0.96

(b) Final category-based results when $\alpha = 0.35$. Average number of iterations is 9.0.

	A	B	C	D	E	F	G
A	0.96	0.96	0.94	0.9	0.85	0.81	0.77
B	0.96	0.96	0.94	0.9	0.86	0.82	0.78
C	0.94	0.94	0.95	0.93	0.88	0.85	0.8
D	0.9	0.91	0.93	0.94	0.92	0.88	0.85
E	0.85	0.86	0.88	0.92	0.94	0.93	0.89
F	0.81	0.81	0.84	0.88	0.92	0.93	0.92
G	0.76	0.76	0.79	0.84	0.88	0.91	0.94

(c) Final category-based results when $\alpha = 0.7$. Average number of iterations is 23.96.

	A	B	C	D	E	F	G
A	0.89	0.88	0.82	0.72	0.62	0.57	0.55
B	0.88	0.88	0.83	0.73	0.62	0.58	0.55
C	0.82	0.83	0.84	0.79	0.67	0.62	0.59
D	0.73	0.73	0.8	0.81	0.76	0.69	0.65
E	0.62	0.63	0.67	0.76	0.79	0.78	0.73
F	0.57	0.58	0.62	0.68	0.78	0.79	0.79
G	0.54	0.54	0.58	0.64	0.72	0.79	0.84

(d) Final category-based results when $\alpha = 0.95$. Average number of iterations is 102.0.

Figure 9: Probability of choosing correct pairs between categories when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, and $\beta_t = 4$ for $\alpha = 0.05, 0.35, 0.7,$ and 0.95.

Different α values and therefore different configurations of the $W_{t\to\infty}$ matrix result in different testing performance. In Figure 9, we report the testing results for $\alpha = 0.05, 0.35, 0.7, 0.95$ when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, and $\beta_t = 4$.

We observe that the probabilities of choosing correct relations in Figures 9c and 9d, respectively, for $\alpha = 0.05$ and $\alpha = 0.35$ are almost the same. In Figure 9a, when $\alpha = 0.7$, the transitive and equivalence relations are affected negatively. In Figure 9d, we see from the converged transition matrix that values for all the relations have decreased. Moreover, for smaller values of α, the convergence of the network needs fewer iterations; compare 4, 9, 23, and 102 for, respectively, $\alpha = 0.05, 0.35, 0.7,$ and 0.95. For more details in α parameter effect, see Table 4, where the connection weights of AB and AG in $W_{t\to\infty}$ for different α choices, along with the calculated probabilities based on three choices of $\beta_t = 1, 4, 8$, are reported.

Table 4: The Simultaneous Effect of α and β_t Values on the Test Results for AB and AG Relations.

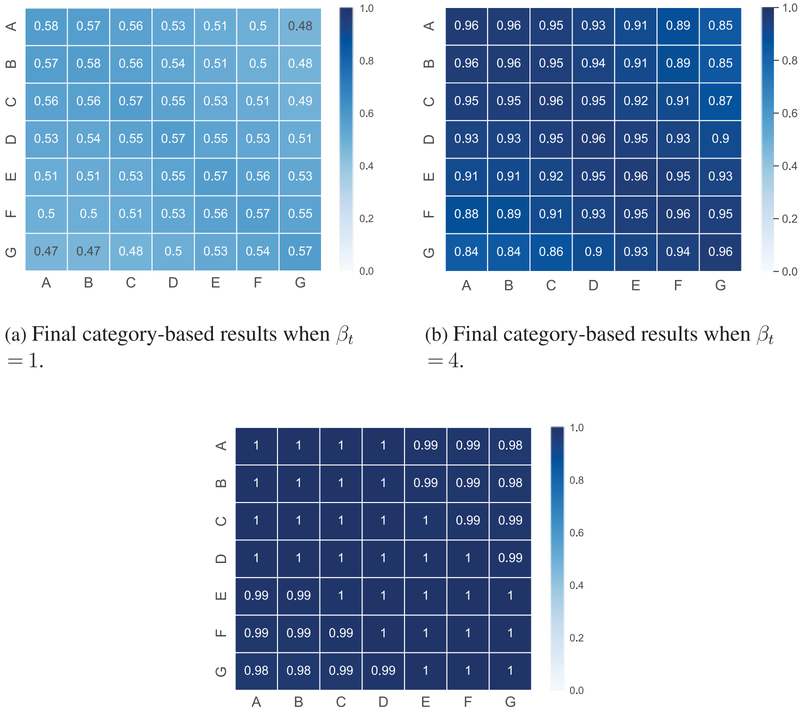
		Baseline Relation AB			Derived Relation AG		
(α, β_t)		A_1B_1	A_1B_2	A_1B_3	A_1G_1	A_1G_2	A_1G_3
$\alpha = 0$	$W_{t\rightarrow\infty}$	0.49837	0.00163	0.00163	0	0	0
	$W_{t\rightarrow\infty_C}$	0.99350	0.00325	0.00325	0	0	0
	$\beta_t = 1$	0.57134	0.21419	0.21447	0.33333	0.33333	0.33333
	$\beta_t = 4$	0.9619	0.01904	0.01906	0.33333	0.33333	0.33333
	$\beta_t = 8$	0.99925	0.00037	0.00037	0.33333	0.33333	0.33333
$\alpha = 0.05$	$W_{t\rightarrow\infty}$	0.49686	0.0017	0.0017	$4.1276e^{-08}$	$5.6875e^{-09}$	$8.6627e^{-09}$
	$W_{t\rightarrow\infty_C}$	0.9932	0.0034	0.0034	0.74202	0.10225	0.15573
	$\beta_t = 1$	0.57115	0.21429	0.21456	0.48349	0.25909	0.25743
	$\beta_t = 4$	0.96178	0.01909	0.01912	0.83865	0.08146	0.07989
	$\beta_t = 8$	0.99925	0.00037	0.00037	0.99049	0.00509	0.00442
$\alpha = 0.9$	$W_{t\rightarrow\infty}$	0.39782	0.02119	0.0223	0.003	0.00092	0.00112
	$W_{t\rightarrow\infty_C}$	0.90145	0.04802	0.05053	0.59524	0.18254	0.22222
	$\beta_t = 1$	0.51983	0.24069	0.23948	0.4146	0.29794	0.28746
	$\beta_t = 4$	0.91757	0.04108	0.04135	0.69558	0.17058	0.13384
	$\beta_t = 8$	0.99726	0.00137	0.00136	0.96297	0.02154	0.01549
$\alpha = 0.95$	$W_{t\rightarrow\infty}$	0.34433	0.03844	0.04031	0.00464	0.00185	0.00212
	$W_{t\rightarrow\infty_C}$	0.81387	0.090858	0.095278	0.53891	0.21487	0.24623
	$\beta_h = 1$	0.47784	0.2627	0.25945	0.39334	0.31058	0.29608
	$\beta_h = 4$	0.85289	0.0733	0.0738	0.61673	0.22268	0.16059
	$\beta_h = 8$	0.99183	0.0041	0.00407	0.92776	0.04397	0.02827

Notes: The $W_{t\rightarrow\infty}$ row reports the weights in the converged network. $W_{t\rightarrow\infty_C}$ refers to the input weights conditioned based on the category that softmax function uses to generate the probability distribution. The C in the index of $W_{t\rightarrow\infty_C}$ refers to the conditional weights for the category calculated with Bayes' rule.

4.6 Experiment 6: Effect of the β_t Parameter. To study the effect of β_t, first we keep other parameters fixed ($\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, $\alpha = 0.05$) and simulate the agent behavior for $\beta_t = 1, 4, 8$ (see Figure 10).

We see a decrease in all types of relations by decreasing the value of β_t. In Figure 10a, when $\beta_t = 1$, all relations, including baseline relations, become weaker. When $\beta_t = 4$ in Figure 10b, we see that the relations are well formed across all nodal numbers. Figure 10c shows that with a higher value of $\beta_t = 8$, all the relations are almost completely formed. This experiment illustrates that by changing β_t, one can control the agent performance in the testing phase and even impair the baseline relations. In Table 4, we take a closer look at the simultaneous effect of α and β_t when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$.

In Table 4, baseline relation AB and transitive relation AG with nodal number five are addressed. We use the conditioned weights (row $W_{t\rightarrow\infty_C}$) as the input vector to the softmax function to generate the probability distribution for the test phase. When $\alpha = 0$, there is no NE, and any choice of β_t results in an equal probability of all relations in AG. However, β_t



(a) Final category-based results when $\beta_t = 1$.

(b) Final category-based results when $\beta_t = 4$.

(c) Final category-based results when $\beta_t = 8$.

Figure 10: Probability of choosing correct relations between categories when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.1$, and $\alpha = 0.05$ for $\beta_t = 1, 4, 8$.

could affect the AB relation so that the performance of the agent is very poor (it chooses A_1B_1 with probability 0.57134 for $\beta_t = 1$) or very strong (it chooses A_1B_1 with probability 0.99925 for $\beta_t = 8$). When $\alpha = 0.05$, $W_{t\to\infty}$ is achieved after about just four iterations. We observe an insignificant reduction in the A_1B_1 weight in $W_{t\to\infty}$ (from 0.49837 to 0.49686) and an insignificant increase in A_1B_2, A_1B_3, A_1G_1, A_1G_2, and A_1G_3. Interestingly, since we use conditioned weights and apply a softmax function, very tiny values for AG in $W_{t\to\infty}$ transfer into noticeable values when conditioned, which could show the formation of derived relations. For instance, with $\beta_t = 4$, $(A_1G_1, A_1G_2, A_1G_3)_{W_{t\to\infty}} = (4.1276e^{-08}, 5.6875e^{-09}, 8.6627e^{-09})$ is transformed to (0.74202, 0.10225, 0.15573) and when softmax is used, it is converted into (0.83865, 0.08146, 0.07989), that is, an A_1G_1 relation is formed for the agent. This means a small value of α and, consequently, a few steps of NE could produce the desired network with an appropriate choice of β_t.

Table 5: The Training Order for OTM.

	Number of Trials per Relation					
Training	*AB*	*AC*	*AD*	*AE*	*AF*	*AG*
AB	48					
AC	24	24				
AD	12	12	24			
AE	9	9	9	24		
AF	6	6	6	6	24	
AG	3	3	3	6	9	24
Baseline maintenance	3	3	3	3	3	3

If we consider higher values of α, we see that the weight of baseline relation A_1B_1 in $W_{t\rightarrow\infty}$ is reduced, but all other relations are enhanced.

It is also noteworthy that increasing the value of A_1G_1, which happens with a higher choice of α, is not equivalent to better performance in the testing phase as reported in Table 4.

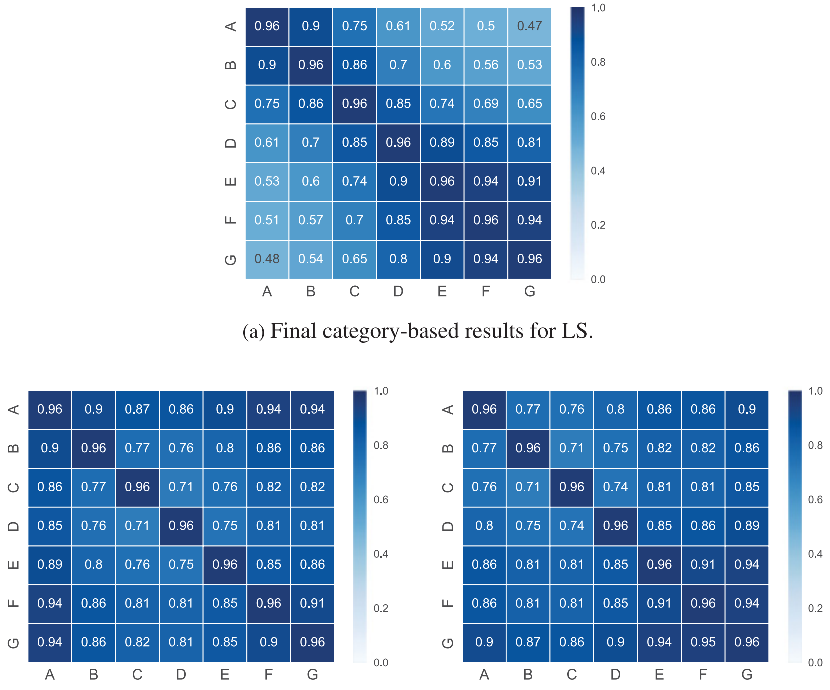
The reason is that NE changes the proportion of weights in $W_{t\rightarrow\infty}$, which affects the conditioned vector in favor of undesired options (see the $W_{t\rightarrow\infty_C}$ values), and, finally, the probability of a correct choice computed through the softmax function is reduced. For instance, when $\alpha = 0.05$, the A_1G_1 weight is $4.1276e^{-08}$, but its proportion in the conditioned vector is 0.74202. For $\alpha = 0.95$, the A_1G_1 weight is 0.00464, which is much higher than $\alpha = 0.05$, but its proportion in the conditioned vector is 0.53891, which is less than the case with $\alpha = 0.05$. So different configurations of α and β_t could produce different behaviors on request.

4.7 Experiment 7: Studying the Training Order: Comparing LS, MTO, and OTM. There are many studies on the differences between LS, OTM, and MTO training structures (see, e.g., Arntzen et al., 2010; Arntzen & Hansen, 2011; Arntzen, 2012). In this experiment, we rearrange the training blocks from LS in Table 1 to similar training stages for OTM and MTO training structures, represented in Tables 5 and 6, respectively. For the OTM training structure, the training relations in order are AB, AC, AD, AE, AF, and AG. For the MTO training structure, the training relations in order are AG, BG, CG, DG, EG, and FG.

The LS, OTM, and MTO training structures can be studied in various levels and with several parameter assemblies. But the aim of this experiment is to show the potential of the proposed E-EPS model in reflecting the differences between the LS, OTM, and MTO training structures reported in the literature. Figure 11 reports the results of the final testing phase of the three cases for an agent with parameters $\gamma = 0.001$, $K = 1$, $\beta_h = 0.05$, $\alpha = 0.05$, and $\beta_t = 4$.

Table 6: The Training Order for MTO.

	Number of Trials per Relation					
Training	*AG*	*BG*	*CG*	*DG*	*EG*	*FG*
AG	48					
BG	24	24				
CG	12	12	24			
DG	9	9	9	24		
EG	6	6	6	6	24	
FG	3	3	3	6	9	24
Baseline maintenance	3	3	3	3	3	3



(a) Final category-based results for LS.

(b) Final category-based results for OTM.

(c) Final category-based results for MTO.

Figure 11: Probability of choosing the correct relations between categories when $\gamma = 0.001$, $K = 1$, $\beta_h = 0.05$, $\alpha = 0.05$, and $\beta_t = 4$ for LS, MTO, and OTM.

According to Figure 11a, the agent performance when the LS is used is not satisfactory for higher nodal numbers. The weakest value, 0.47, belongs to *AG*. The equivalence classes are not formed in this case. Figure 11b shows better performance where the weakest connections are for *CD* and *DC* and equal 0.71. This minimum value is also found in Figure 11c but for relations *BC* and *CB*. So in this experiment, the overall results in terms of formation of equivalence classes are the same for MTO and OTM, but due to the order of training, the agent might exhibit different performance for specific relations in MTO and OTM training structures. For instance, the calculated probability for an *FA* relation in OTM is 0.94, and in MTO it is 0.86. Calculated probability for the *DE* relation in OTM is 0.75, while in MTO it is 0.85.

It is noteworthy that the training times—that is, the numbers of repetitions of each block before mastery in all three cases for all training procedures—are similar. This can be explained by the independence of designing baseline relations. The reported results in Figure 11 confirm that our model shows better performance in the OTM and MTO cases in comparison with LS (Arntzen et al., 2010; Arntzen & Hansen, 2011; Arntzen, 2012).

5 Conclusion

The main contribution of this letter is to offer a new perspective in the formation of SE classes in a recently introduced model, EPS. EPS is a modified version of the PS model (Briegel & De las Cuevas, 2012) and can be seen as an RL agent that has a directed, weighted network of clips. Each clip represents a remembered stimulus that is added to the clip network during the training phase.

To replicate the test phase of SE by examining the agent's ability to encounter new relations that can be derived from baseline relations, the EPS model relies on some type of likelihood reasoning whenever tested via an MTS trial. In other words, in the EPS model, derived relations were calculated on demand in the testing phase trials, but the new approach to the testing phase is offline and relies on memory retrieval during the testing phase rather than on complex logical processing. Derived relations in the new model, E-EPS, are achieved by applying an iterative diffusion process, network enhancement (NE; Wang et al., 2018). During the NE phase, the structure of the clip network changes where indirect relations get enhanced. The NE is a denoising method, and one way to interpret the model is to consider a typical memory as a less noisy memory, while a disabled memory is a noisy memory that cannot form equivalence relations. Since in the NE, connections are bidirectional, we refer to it is as symmetric network enhancement (SNE) in this letter. We further modify the SNE and propose directed network enhancement (DNE) in which the connections are directed and where we can control the agent's ability to derive transitivity and symmetry. One might use SNE in studying SE formation with the assumption that all the relations are bidirectional and transitive and equivalence relations are formed. DNE is a better option to replicate real experiments with the possibility of nonformation of classes and nonsymmetric relations.

In the simulation part, we study the role of parameters on agent performance and show that the model is able to replicate either a typical memory or a disabled memory with different learning and forgetting rates and accomplish the trial tasks in the testing phase. We also compare the main training structures, LS, MTO, and OTM, and notice the better outcome of MTO and OTM training structures than that of LS, which is consistent with evidence from the behavioral analysis literature. Many other configurations can be considered in simulations. For instance, we consider $K = 1$ to reduce the variety of results, and to study each parameter, we fixed all the other parameters.

Another alternative is to execute the NE phase during training rather than merely at the end of the training. The argument would be that brain does not wait until the end of training to start the process of formation of these relations. Although this might sound a like plausible argument and can be easily added to the model, we avoid NE during training. The most obvious reason is to keep the model simple, with fewer computations. Because we are studying agent behavior, the timing of events inside the brain is not our priority. Moreover, baseline relations are independent and not derived from each other, so there is no need to update them earlier when the formation of relations is tested in the testing phase. However, as discussed in section 3.1, these updates could be analogous to the replay in the brain that generates a predictive map in an offline process.

The probability distribution over comparison stimuli in the test trial is calculated based on the direct links in the updated clip network. It is similar to the EPS in the sense that whenever there are links between the sample stimulus and comparison stimuli, the probabilities are calculated based on the h-values by averaging or using a softmax function. In E-EPS, however, there are links through the entire network updated by the NE process, and therefore no extra calculation is made. Although one might still consider the random walk on the network similar to the PS model, the cyclic nature of the network in E-EPS might generate problems, and extra conditions (such as gating) might be necessary. We avoid this scenario, since the calculated weights are based on the random walk and diffusion, and we consider these cached links at the decision time. The EPS and E-EPS could be developed further to model more complex tasks with more sophisticated structures as the PS model offers. For instance, we might use compound stimuli and benefit from a PS model with associative learning (Briegel & De las Cuevas, 2012), or a multilayer memory clip where the agent is able to generate and add wildcard to the memory (Melnikov et al., 2017). Such multi-layer PS agent has been further developed to address abstract compositional concepts which is closer to the concept of SE (Ried, Eva, Müller, & Briegel, 2019). The mathematical understanding of the properties of the converged network that guarantees the converged solution is an advantage of NE over other network denoising methods. DNE maintains many properties of SNE with the advantage of controlling the formation of symmetry and transitivity in the E-EPS model. Finally, it is worth mentioning that we choose NE as the source of inspiration for updating the network clip, since in the updates, there is no requirement for supervision or prior knowledge. After the training phase, we have a clip network without further feedback or supervision. Hence, NE provides a proper solution with an emphasis on the indirect paths, which is what we have in derived relations.

Appendix A: Theoretical Analysis of Directed Network Enhancement

In this appendix, we explain why the proposed diffusion process in equation 3.1 improves the results and can be used to form equivalence classes. Our theoretical analysis is mostly based on supplementary note 3 of Wang et al. (2018). However, since W_t in the DNE is not a symmetric doubly stochastic matrix, the proofs and discussions need to be revised for DNE. It is noteworthy that the largest eigenvalue of each right stochastic matrix, such as P, is 1, associated with eigenvector $\mathbf{1}$. We first prove that W_t remains right stochastic in each iteration of DNE and converges to a nontrivial equilibrium matrix. Then we show that DNE preserves the eigenvectors of the stochastic matrix W_0, but increases the gap between large eigenvalues and reduces the gap between small eigenvalues (see Figure 13). The larger eigengap in the final converged matrix $W_{t\to\infty}$, is associated with better equivalence class formation.

A.1 The Convergence of the DNE Process.

We show that W_t remains stochastic during the updates. By definition $W_0\mathbf{1} = \mathbf{1}$, for all-one eigenvector $\mathbf{1}$ associated with eigenvalue 1. We assume that $W_{t-1}\mathbf{1} = \mathbf{1}$ and show that the rows of W_t remain normalized:

$$\begin{aligned} W_t\mathbf{1} &= \alpha P W_{t-1} P\mathbf{1} + (1-\alpha)P\mathbf{1} \\ &= \alpha P W_{t-1}\mathbf{1} + (1-\alpha)P\mathbf{1} \\ &= \alpha P\mathbf{1} + (1-\alpha)P\mathbf{1} \\ &= P\mathbf{1} \\ &= \mathbf{1}. \end{aligned} \tag{A.1}$$

Now we show that W_t converges to a nontrivial equilibrium graph. A closed-form solution for the final, converged network can be achieved through induction. Consider the following expression for the network at iteration t:

$$W_t = \alpha^t P^t W_0 P^t + (1-\alpha)P\sum_{k=0}^{t-1}(\alpha P^2)^k. \tag{A.2}$$

This formula is similar to equation 6 of supplementary note 3 by Wang et al. (2018) where $\mathcal{T}$ is replaced by P and can be guessed by iterating the process for the first few steps:

1. Define $W_0 = W_{t=0}$. For $t = 1$, equation A.2 holds true:

$$W_{t=1} = \alpha P W_0 P + (1-\alpha)P$$

2. We assume equation A.2 holds true for iteration t. Then:

$$\begin{aligned} W_{t+1} &= \alpha P W_t P + (1-\alpha)P \\ &= \alpha P \left(\alpha^t P^t W_0 P^t + (1-\alpha)P \sum_{k=0}^{t-1} (\alpha P^2)^k \right) P + (1-\alpha)P \\ &= \alpha^{t+1} P^{t+1} W_0 P^{t+1} + (1-\alpha)P \sum_{k=0}^{t-1} (\alpha P^2)^{k+1} + (1-\alpha)P \\ &= \alpha^{t+1} P^{t+1} W_0 P^{t+1} + (1-\alpha)P \sum_{k=0}^{t} (\alpha P^2)^k, \end{aligned}$$

which satisfies equation A.2. Using geometric series when $t \to \infty$, we have this nontrivial equilibrium matrix:

$$W_{t\to\infty} = (1-\alpha)P(\mathcal{I} - \alpha P^2)^{-1}. \tag{A.3}$$

A.2 Spectral Analysis of DNE. We show that the DNE process does not change eigenvectors of the input matrix $W_0 = P$ but maps eigenvalues through a nonlinear function.

Suppose (λ, v) is the eigenpair of P. We know that the absolute value of eigenvalues of any stochastic matrix satisies the $|\lambda| \leq 1$ relation. Let the eigendecomposition of P be VDV^{-1}, where D is a diagonal matrix formed from eigenvalues of P and the columns of V are the corresponding eigenvectors of P. We have

$$\begin{aligned} W_{t\to\infty} &= (1-\alpha)P(\mathcal{I} - \alpha P^2)^{-1} \\ &= (1-\alpha)VDV^{-1}(\mathcal{I} - \alpha VDV^{-1}VDV^{-1})^{-1} \\ &= (1-\alpha)VDV^{-1}(VV^{-1} - \alpha VDV^{-1}VDV^{-1})^{-1} \\ &= (1-\alpha)VDV^{-1}\left(V(\mathcal{I} - \alpha D^2)V^{-1}\right)^{-1} \\ &= (1-\alpha)VDV^{-1}\left(V(\mathcal{I} - \alpha D^2)^{-1}V^{-1}\right) \\ &= (1-\alpha)V\left(D(\mathcal{I} - \alpha D^2)^{-1}\right)V^{-1} \\ &= V\left((1-\alpha)(D(\mathcal{I} - \alpha D^2)^{-1}\right)V^{-1} \\ &= VD'V^{-1}. \end{aligned}$$

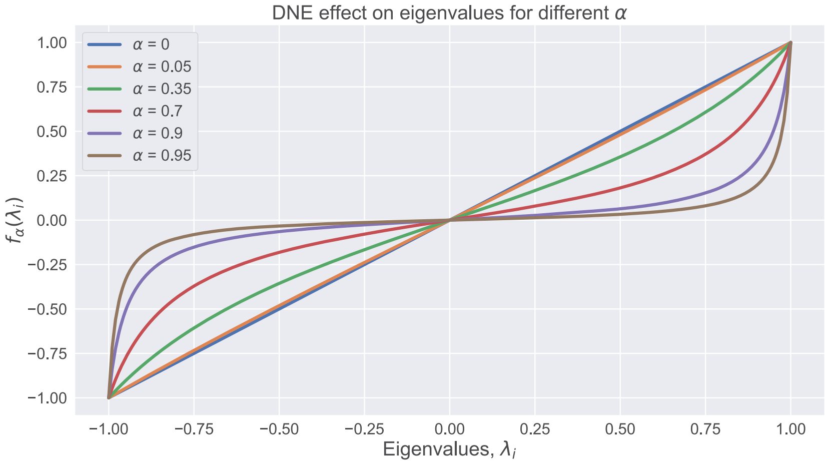


Figure 12: Role of α on the nonlinear transformation of eigenvalues using $f_\alpha(\lambda)$ in the DNE process.

This testifies that the DNE process keeps the eigenvectors unchanged, but the eigenvalues become $D'_{ii} = \frac{(1-\alpha)\lambda_i}{1-\alpha\lambda_i^2}$. Therefore, the DNE process functions nonlinearly on the eigenvalues of the input matrix, that is, the final converged matrix, $W_{t\to\infty}$, transforms (λ, υ) to $(f_\alpha(\lambda), \upsilon)$, where $f_\alpha(\lambda) = \frac{(1-\alpha)\lambda}{1-\alpha\lambda^2}$. It is trivial that $f_\alpha(\lambda)(0) = 0$, $f_\alpha(\lambda)(1) = 1$. The following relations show that the DNE always decreases the absolute value of eigenvalues,

$$\begin{aligned}
1 &\geq |\lambda|, \\
1 &\geq \lambda^2, \\
\alpha &\geq \alpha\lambda^2, \\
1-\alpha &\leq 1-\alpha\lambda^2, \\
|\lambda|(1-\alpha) &\leq |\lambda|(1-\alpha\lambda^2), \\
\frac{|\lambda|(1-\alpha)}{1-\alpha\lambda^2} &\leq |\lambda|,
\end{aligned}$$

where the rate of this decrease is higher for eigenvalues with greater absolute values. Figure 12 depicts the behavior of f_α and how this nonlinear function can be regularized with an α parameter. Increasing the eigengaps between large eigenvalues enhances the robustness of the converged network, which in our case means a better formation of classes (for details on the spectral eigengap, see Joseph & Yu, 2016; Wang et al., 2018; Mavroeidis & Bingham, 2010).

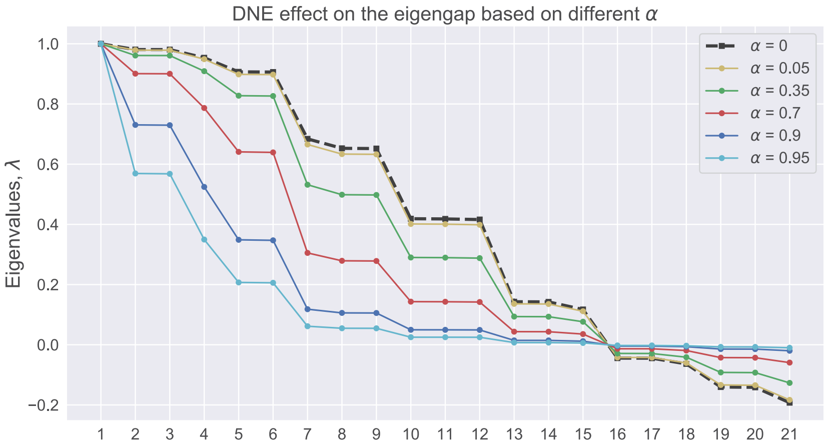


Figure 13: The effect of α on the eigenvalues of the transition matrix of a clip network obtained from experiment 1 in section 4 (see Table 1 for the training structure).

Figure 12 shows that by increasing the regularization parameter, higher eigengaps are achieved. In Figure 13, the associated eigenvalues of a sample network clip[7] and the new eigenvalues of the converged network with different α values are represented.

Appendix B: Training Structure Design Complexity

Here we provide some mathematical calculations to show how complex the design of different training structures could be in real experiments and artificial EPS or E-EPS agents.

Let the set of all classes be $\mathbf{C}$, where each class has m members. Each member of the classes belongs to a separate category, usually labeled by letters A, B, C, and so on. As a result, there are m categories, each with $n = |\mathbf{C}|$ members, so the total number of stimuli equals $m|\mathbf{C}| = mn$. In an arbitrary MTS procedure, the experimenter usually decides how to label categories (among $m!$ possibilities) and which stimuli sets form classes (among $mn!$ possibilities). In real-life experiments, changing the order of two categories (or labels) or how the members of the same class are assembled across different categories might have an impact on the learning and testing outcome.

However, in the computational model, all the categories and stimuli are abstract symbols and are literally the same. We just use the category labels and class indices to differentiate the stimuli. When there is differentiation

[7] The training order is represented in Table 1, and the experiment is clarified in section 4.

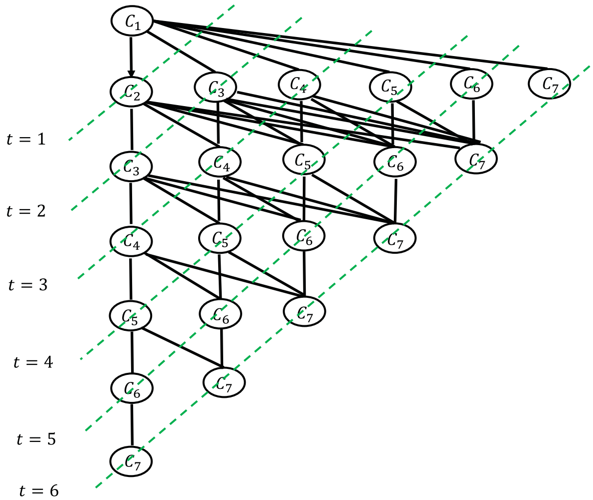


Figure 14: C_1 to C_7 refers to the seven categories and the number of possible maps from categories to C_is, $i = 1, \cdots 7$ is 7!. At each time step, shown by green dashed lines, a category is added to the previously trained relations. At time $t = 1$, the C_1 to C_2 relation, which is shown via a directed connection, is trained as the first relation. This can be any relation. Then at each time step, a new category is connected to the previously trained relations.

between categories in a real-life experiment, the total number of baseline relation configurations, defined as **T**, would be

$$\mathbf{T} = \binom{m}{1}\binom{m-1}{1}\left(2\binom{2}{1}\binom{m-2}{1}\right)\left(2\binom{3}{1}\binom{m-3}{1}\right)\cdots\left(2\binom{m-1}{1}\binom{1}{1}\right)$$
$$= 2^{m-2}m!(m-1)! \tag{B.1}$$

In the EPS model, we can remove the repetitions by assuming the category label describes the order of adding a category. For instance, the first relation for training would be AB, the next training could be one of AC, BC, CA or CB, and so on. The number of different training configurations for the agent in this case is

$$\mathbf{T} = 1 \times \left(2\binom{2}{1}\right) \times \left(2\binom{3}{1}\right) \cdots \left(2\binom{m-1}{1}\right) = 2^{m-2}(m-1)! \tag{B.2}$$

To make these calculations more intuitive, consider the case with seven categories, that is, $m = 7$, with labels $A, B, C, D, E, F,$ and, G, each with three members $n = 3$. In Figure 14, C_1 to C_7 refers to the seven categories where

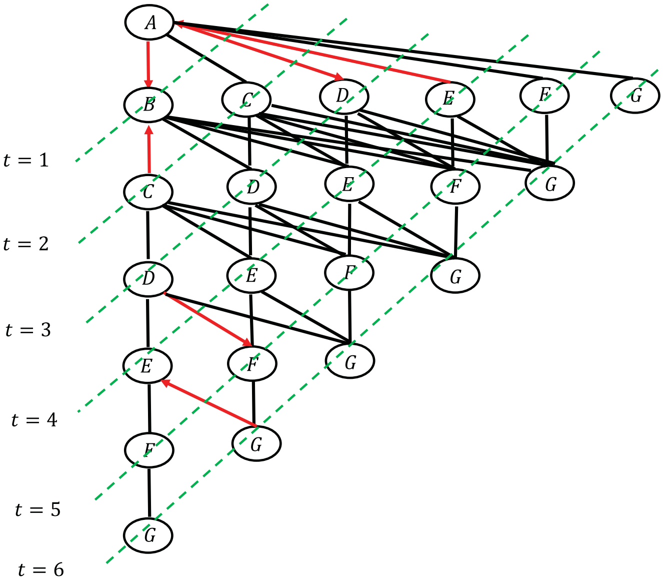
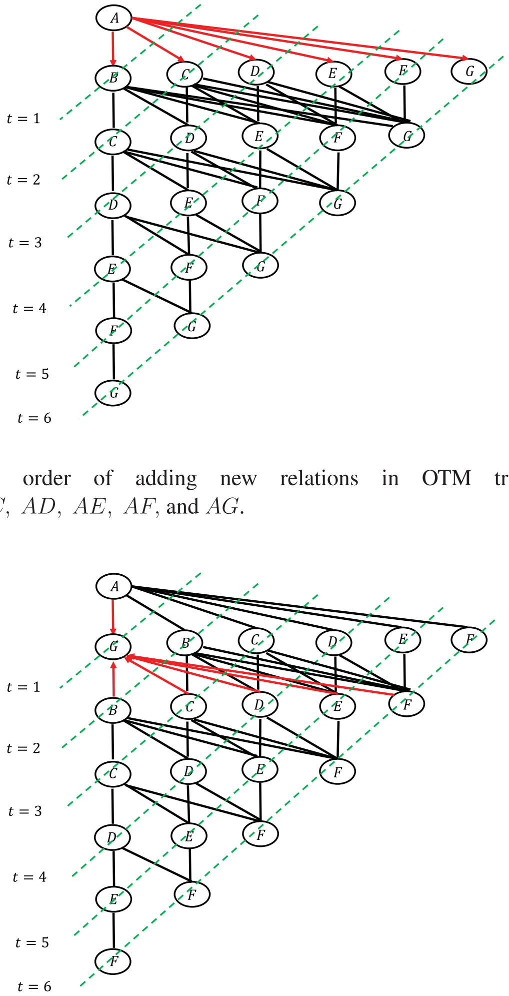


Figure 15: A possible training structure is shown in red—*AB*, *CB*, *AD*, *EA*, *DF*, *GE*—when the order of categories in the training structure is not important.

at each time step, one relation to a new category will be added. The first training stage contains the C_1 to C_2 relation, which is shown via a directed connection. C_1 could be any of seven categories, and C_2 could be one of the remaining six categories. The next stage, represented with $t = 2$ is to add C_3, which is one of the remaining five categories. There are four options to train: C_1C_3, C_3C_1, C_2C_3, and C_3C_2, shown with undirected connections. Similarly, we see that for $t = 3$, there are four choices for categories and 2×3 ways to choose the relation that connects C_4 to previous categories. Therefore, we can easily see that the number of possible maps of categories to C_1 to C_7 is 7! and the possibility them with six relations is $2^5(6!)$. In total, if we distinguish between categories and therefore their order, the number of possible training procedures based on equation B.1 and our explanation equals $2^5(7!)(6!) = 32 \times 5040 \times 720 = 116, 121, 600$.

If we consider the order of categories to be the same and map $C_1 \rightarrow A$, $C_2 \rightarrow B$, $C_3 \rightarrow C$, $C_4 \rightarrow D$, $C_5 \rightarrow E$, $C_6 \rightarrow F$, and $C_7 \rightarrow G$, different configurations will be reduced to $2^5(6!) = 32 \times 720 = 23{,}040$, according to equation B.2. This one-to-one mapping is shown in Figure 15, along with a sample training order in directed red connections that is not LS, OTM, or MTO (see Table 7 for a summary of the training).



(a) The order of adding new relations in OTM training structure: AB, AC, AD, AE, AF, and AG.

(b) The order of adding new relations in MTO training structure: AG, BG, CG, DG, EG, and FG.

Figure 16: Graphical representation of training order for OTM and MTO, shown in red.

In Figures 16a and 16b, respectively, the order of adding new relations to the training blocks for OTM and MTO is depicted. Both training structures are addressed in experiment 1 and reported in Tables 5 and 6.

Although our argument and equations B.1 and B.2 show the complexity of studying the effect of a training structure in an MTS procedure on the

Table 7: Training Order for the Training Structure Depicted in Figure 15.

Time Step	New Relation	Possible Previous Relations				
$t = 1$	AB					
$t = 2$	CB	AB				
$t = 3$	AD	CB	AB			
$t = 4$	EA	AD	CB	AB		
$t = 5$	DF	EA	AD	CB	AB	
$t = 6$	GE	DF	EA	AD	CB	AB

Note: A training block can be formed by only new relation at each stage or a combination of new and previously trained relations.

participant/agent performance, the training structure and training block design are much more complex. We have addressed the order of adding new training relation to the previously trained relations. Many other parameters can be included in the analysis, such as the number of trials in each block, the combination of previously trained relations together with the new relation, testing derived relations during training or not, testing order, and number of classes (members of each category). Moreover, the possibility of training a mixture of relations between two categories, say, A_1B_1, B_2A_1, A_3B_3, will increase this number. An example of such training is simulated in our previous work (Mofrad et al., 2020). Therefore, finding some optimal training structure either theoretically or via simulation with EPS or E-EPS is an interesting problem in its own right, but it is out of the scope of this letter.

Abbreviations

DNE	Directed Network Enhancement.
DSM	doubly stochastic matrix.
E-EPS	Enhanced Equivalence Projective Simulation.
EPS	Equivalence Projective Simulation.
fMRI	Functional Magnetic Resonance Imaging.
LS	linear series.
MTO	many-to-one.
MTS	matching-to-sample.
NE	Network Enhancement.
OTM	one-to-many.
PS	Projective Simulation.
RL	reinforcement learning.
SE	Stimulus Equivalence.
SNE	Symmetric Network Enhancement.

Acknowledgments

We thank the anonymous reviewers for their constructive and thorough feedback that helped us improve the quality of this letter. The source code of the E-EPS model simulator is made available online at https://github.com/Asieh-A-Mofrad/Enhanced-Equivalence-Projective-Simulation.

References

Arntzen, E. (2012). Training and testing parameters in formation of stimulus equivalence: Methodological issues. *European Journal of Behavior Analysis*, *13*(1), 123–135.

Arntzen, E., Grondahl, T., & Eilifsen, C. (2010). The effects of different training structures in the establishment of conditional discriminations and subsequent performance on tests for stimulus equivalence. *Psychological Record*, *60*(3), 437–461.

Arntzen, E., & Hansen, S. (2011). Training structures and the formation of equivalence classes. *European Journal of Behavior Analysis*, *12*(2), 483–503.

Arntzen, E., & Holth, P. (1997). Probability of stimulus equivalence as a function of training design. *Psychological Record*, *47*(2), 309–320.

Arntzen, E., & Mensah, J. (2020). On the effectiveness of including meaningful pictures in the formation of equivalence classes. *Journal of the Experimental Analysis of Behavior*, *113*(2), 305–321.

Barnes, D., & Hampson, P. J. (1993). Stimulus equivalence and connectionism: Implications for behavior analysis and cognitive science. *Psychological Record*, *43*(4), 617–638.

Baumeister, R. F., Bratslavsky, E., Finkenauer, C., & Vohs, K. D. (2001). Bad is stronger than good. *Review of General Psychology*, *5*(4), 323–370.

Behrens, T. E., Muller, T. H., Whittington, J. C., Mark, S., Baram, A. B., Stachenfeld, K. L., & Kurth-Nelson, Z. (2018). What is a cognitive map? Organizing knowledge for flexible behavior. *Neuron*, *100*(2), 490–509.

Briegel, H. J., & De las Cuevas, G. (2012). Projective simulation for artificial intelligence. *Scientific Reports*, *2*(1), 1–16.

Cullinan, V. A., Barnes, D., Hampson, P. J., & Lyddy, F. (1994). A transfer of explicitly and nonexplicitly trained sequence responses through equivalence relations: An experimental demonstration and connectionist model. *Psychological Record*, *44*(4), 559–585.

Daw, N. D., Gershman, S. J., Seymour, B., Dayan, P., & Dolan, R. J. (2011). Model-based influences on humans' choices and striatal prediction errors. *Neuron*, *69*(6), 1204–1215.

Daw, N. D., Niv, Y., & Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature Neuroscience*, *8*(12), 1704–1711.

Devany, J. M., Hayes, S. C., & Nelson, R. O. (1986). Equivalence class formation in language-able and language-disabled children. *Journal of the Experimental Analysis of Behavior*, *46*(3), 243–257.

Fields, L., Adams, B. J., Verhave, T., & Newman, S. (1990). The effects of nodality on the formation of equivalence classes. *Journal of the Experimental Analysis of Behavior*, *53*(3), 345–358.

Fienup, D. M., Wright, N. A., & Fields, L. (2015). Optimizing equivalence-based instruction: Effects of training protocols on equivalence class formation. *Journal of Applied Behavior Analysis*, *48*(3), 613–631.

Garvert, M. M., Dolan, R. J., & Behrens, T. E. (2017). A map of abstract relational knowledge in the human hippocampal–entorhinal cortex. *Elife*, *6*, e17086.

Groskreutz, N. C., Karsina, A., Miguel, C. F., & Groskreutz, M. P. (2010). Using complex auditory-visual samples to produce emergent relations in children with autism. *Journal of Applied Behavior Analysis*, *43*(1), 131–136.

Hayes, S. C. (1989). Nonhumans have not yet shown stimulus equivalence. *Journal of the Experimental Analysis of Behavior*, *51*(3), 385–392.

Hove, O. (2003). Differential probability of equivalence class formation following a one-to-many versus a many-to-one training structure. *Psychological Record*, *53*(4), 617–634.

Joseph, A., & Yu, B. (2016). Impact of regularization on spectral clustering. *Annals of Statistics*, *44*(4), 1765–1791.

Kumaran, D., & McClelland, J. L. (2012). Generalization through the recurrent interaction of episodic memories: A model of the hippocampal system. *Psychological Review*, *119*(3), 573–616.

Lew, S. E., & Zanutto, S. B. (2011). A computational theory for the learning of equivalence relations. *Frontiers in Human Neuroscience*, *5*, 113.

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, *8*(3–4), 293–321.

Liu, T.-Y., & Watson, B. O. (2020). Patterned activation of action potential patterns during offline states in the neocortex: Replay and non-replay. *Philosophical Transactions of the Royal Society B*, *375*(1799), 20190233.

Lyddy, F., & Barnes-Holmes, D. (2007). Stimulus equivalence as a function of training protocol in a connectionist network. *Journal of Speech and Language Pathology–Applied Behavior Analysis*, *2*(1), 14.

Lyddy, F., Barnes-Holmes, D., & Hampson, P. J. (2001). A transfer of sequence function via equivalence in a connectionist network. *Psychological Record*, *51*(3), 409–428.

Mautner, J., Makmal, A., Manzano, D., Tiersch, M., & Briegel, H. J. (2015). Projective simulation for classical learning agents: A comprehensive investigation. *New Gener. Comput.*, *33*(1), 69–114.

Mavroeidis, D., & Bingham, E. (2010). Enhancing the stability and efficiency of spectral ordering with partial supervision and feature selection. *Knowledge and Information Systems*, *23*(2), 243–265.

McClelland, J. L. (2009). The place of modeling in cognitive science. *Topics in Cognitive Science*, *1*(1), 11–38.

McClelland, J. L. (2013). Integrating probabilistic models of perception and interactive neural networks: A historical and tutorial review. *Frontiers in Psychology*, *4*, 503.

Melnikov, A. A., Makmal, A., Dunjko, V., & Briegel, H. J. (2017). Projective simulation with generalization. *Scientific Reports*, *7*(1), 14430.

Mofrad, A. A., Yazidi, A., Hammer, H. L., & Arntzen, E. (2020). Equivalence projective simulation as a framework for modeling formation of stimulus equivalence classes. *Neural Computation*, *32*(5), 912–968.

Momennejad, I. (2020). Learning structures: Predictive representations, replay, and generalization. *Current Opinion in Behavioral Sciences*, *32*, 155–166.

Momennejad, I., Otto, A. R., Daw, N. D., & Norman, K. A. (2017). *Offline replay supports planning: FMRI evidence from reward revaluation*. bioRxiv:196758.

Momennejad, I., Russek, E. M., Cheong, J. H., Botvinick, M. M., Daw, N. D., & Gershman, S. J. (2017). The successor representation in human reinforcement learning. *Nature Human Behaviour*, *1*(9), 680–692.

Ninness, C., Ninness, S. K., Rumph, M., & Lawson, D. (2018). The emergence of stimulus relations: Human and computer learning. *Perspectives on Behavior Science*, *41*(1), 121–154.

O'Keefe, J., & Nadel, L. (1978). *The hippocampus as a cognitive map*. Oxford: Clarendon Press.

O'Mara, H. (1991). Quantitative and methodological aspects of stimulus equivalence. *Journal of the Experimental Analysis of Behavior*, *55*(1), 125–132.

Parr, T., Markovic, D., Kiebel, S. J., & Friston, K. J. (2019). Neuronal message passing using mean-field, Bethe, and marginal approximations. *Scientific Reports*, *9*(1), 1–18.

Ratcliff, R., Smith, P. L., Brown, S. D., & McKoon, G. (2016). Diffusion decision model: Current issues and history. *Trends in Cognitive Sciences*, *20*(4), 260–281.

Ried, K., Eva, B., Müller, T., & Briegel, H. J. (2019). *How a minimal learning agent can infer the existence of unobserved variables in a complex environment*. arXiv:1910.06985.

Russek, E. M., Momennejad, I., Botvinick, M. M., Gershman, S. J., & Daw, N. D. (2017). Predictive representations can link model-based reinforcement learning to model-free mechanisms. *PLOS Computational Biology*, *13*(9), e1005768.

Schwöbel, S., Kiebel, S., & Marković, D. (2018). Active inference, belief propagation, and the Bethe approximation. *Neural Computation*, *30*(9), 2530–2567.

Shrager, J., Hogg, T., & Huberman, B. A. (1987). Observation of phase transitions in spreading activation networks. *Science*, *236*(4805), 1092–1094.

Sidman, M. (1971). Reading and auditory-visual equivalences. *Journal of Speech, Language, and Hearing Research*, *14*(1), 5–13.

Sidman, M. (1990). Equivalence relations: Where do they come from? In D. E. Blackman & H. Lejeune (Eds.), *Behaviour analysis in theory and practice: Contributions and controversies* (pp. 93–114). Mahwah, NJ: Erlbaum.

Sidman, M. (1994). *Equivalence relations and behavior: A research story.* Authors Cooperative.

Sidman, M., Cresson Jr., O., & Willson-Morris, M. (1974). Acquisition of matching to sample via mediated transfer 1. *Journal of the Experimental Analysis of Behavior*, *22*(2), 261–273.

Sidman, M., Rauzin, R., Lazar, R., Cunningham, S., Tailby, W., & Carrigan, P. (1982). A search for symmetry in the conditional discriminations of rhesus monkeys, baboons, and children. *Journal of the Experimental Analysis of Behavior*, *37*(1), 23–44.

Sidman, M., & Tailby, W. (1982). Conditional discrimination vs. matching to sample: An expansion of the testing paradigm. *Journal of the Experimental Analysis of Behavior*, *37*(1), 5–22.

Sidman, M., Willson-Morris, M., & Kirk, B. (1986). Matching-to-sample procedures and the development of equivalence relations: The role of naming. *Analysis and intervention in Developmental Disabilities*, *6*(1–2), 1–19.

Spencer, T. J., & Chase, P. N. (1996). Speed analyses of stimulus equivalence. *Journal of the Experimental Analysis of Behavior*, *65*(3), 643–659.

Stachenfeld, K. L., Botvinick, M. M., & Gershman, S. J. (2017). The hippocampus as a predictive map. *Nature Neuroscience*, *20*(11), 1643.

Steingrimsdottir, H. S., & Arntzen, E. (2011). Using conditional discrimination procedures to study remembering in an Alzheimer's patient. *Behavioral Interventions*, *26*(3), 179–192.

Stella, F., Baracskay, P., O'Neill, J., & Csicsvari, J. (2019). Hippocampal reactivation of random trajectories resembling Brownian diffusion. *Neuron*, *102*(2), 450–461.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Sutton, R. S., Szepesvári, C., Geramifard, A., & Bowling, M. (2008). Dyna-style planning with linear function approximation and prioritized sweeping. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence* (pp. 528–536). Arlington, VA: AUAI Press.

Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychological Review*, *55*(4), 189–208.

Tovar, Á. E., & Westermann, G. (2017). A neurocomputational approach to trained and transitive relations in equivalence classes. *Frontiers in Psychology*, *8*, 1848.

Wang, B., Pourshafeie, A., Zitnik, M., Zhu, J., Bustamante, C. D., Batzoglou, S., & Leskovec, J. (2018). Network enhancement as a general method to denoise weighted biological networks. *Nature Communications*, *9*(1), 3108.

Wimmer, G. E., & Shohamy, D. (2012). Preference by association: How memory mechanisms in the hippocampus bias decisions. *Science*, *338*(6104), 270–273.

Received March 20, 2020; accepted September 1, 2020.