A COGNITIVE MODEL FOR LEARNING ABSTRACT RELATIONAL STRUCTURES FROM MEMORY-BASED DECISION-MAKING TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Motivated by a recent neuroscientific hypothesis, some theoretical studies have accounted for the neural cognitive maps in the hippocampal formation as a representation of the general relational structure across task environments. However, despite their remarkable results reproducing place and grid cell properties, it is unclear whether their account can be extended to more general settings beyond spatial random-walk tasks in 2D environments. To address this question, we construct a novel cognitive model that performs memory-based decision-making tasks for learning abstract relational structures. Building on recent approaches of modular representation of abstract relations and concrete entities, we develop a learning algorithm that performs reward-guided relational inference across different entity domains, where we adopt a specific memory mechanism with content replacement to maintain dynamic binding between relations and entities. Our experiments show (i) the capability of our model to capture relational structures that can generalize over new domains with unseen entities, (ii) the difficulty of our task that leads existing powerful models, including Neural Turing Machine and vanilla Transformer, to complete failure, and (iii) the similarity of performance and internal representations of our model to recent human behavioral and fMRI experimental data, including distance coding and hexagonal modulation properties in the hippocampal formation.

1 INTRODUCTION

In everyday human cognition, we often find relationships among entities. Sometimes, we discern common relational structures across various domains and lift it to general knowledge (Figure 1). For example, ordering can be found not only among numbers, but also among objects and among individuals. In fact, ordering is so ubiquitous that it would be useful to consider the abstract notion of ordering irrespective of the concrete entities. This is just one example of *abstract relational structure*; other examples include tree-like structure, cyclic structure, and so on.

In recent neuroscience, it has been hypothesized that the hippocampal formation[1] may play a central role in abstract relational representation (Eichenbaum and Cohen, 2014). Indeed, ample evidence exists for relational representations in this neural subsystem (Bunsey and Eichenbaum, 1996; Dusek and Eichenbaum, 1997; Kumaran et al., 2012; Constantinescu et al., 2016; Bao et al., 2019; Park et al., 2020; 2021), some being suggested to be independent of the concrete domain of the entities (Kumaran et al., 2012). On the theoretical side, Whittington et al. (2018; 2020) have remarkably shown, in their proposal of Tolman-Eichenbaum Machine (TEM), that place- and grid-cell properties in the rodent hippocampus and entorhinal cortex (Moser et al., 2008) can emerge as a result of learning the general relational structure of two-dimensional geometry during spatial random-walk tasks. Thus, abstract relational learning is becoming a fascinating, unified view of the hippocampal computation, potentially explaining previously considered multiple functions such as episodic memory and navigation by a single principle.

In this study, we push this direction further by proposing a novel cognitive model that performs memory-based decision-making tasks for learning abstract relational structures. Our tasks, inspired

[1]The hippocampal formation refers to the hippocampus plus its neighbor, the entorhinal cortex.

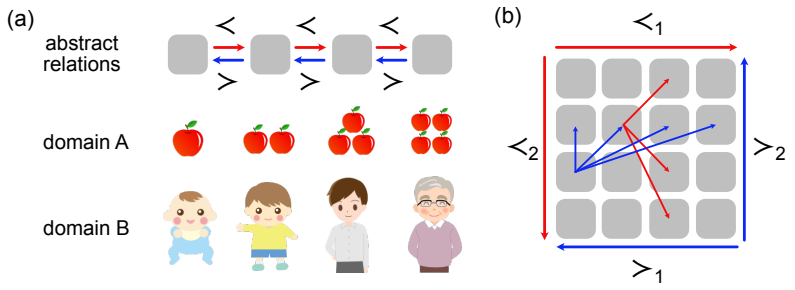


Figure 1: Abstract relational structure. (a) One-dimensional hierarchy (order structure) and its two instantiations for different domains. (b) Two-dimensional hierarchy, with many-to-many relations.

by previous human experiments (Kumaran et al., 2012; Park et al., 2020; 2021), require to repeat pairwise relational inference in one domain after another and eventually learn the common relational structure across domains. To solve the tasks, building on recent approaches of modular architectures for relational representation (Whittington et al., 2018; 2020), we develop a learning algorithm that performs a reward-guided search for disentangled representations of domain-generic relations and domain-specific entities. Importantly, we observe a task nature that requires careful choice of a memory mechanism, a key technique to maintain dynamic binding between relations and entities, which leads us to a specific memory update rule that enables content replacement.

Our experiments yielded the following novel findings. First, our model trained on two types of hierarchical relation successfully learned representations that are generalizable to new domains with completely unknown entities and usable in down-stream tasks such as transitive inference. Second, several existing models such as LSTM (Hochreiter and Schmidhuber, 1997), Neural Turing Machine (NTM) (Graves et al., 2014), Neural Difference Computer (NDC) (Graves et al., 2016), vanilla Transformer (Vaswani et al., 2017), and a version of TEM (Whittington et al., 2018; 2020) could not solve our tasks despite their powerful capabilities. Third, our model exhibited task performance and internal representations that were compatible with behavioral and fMRI data from previous human experiments (Kumaran et al., 2012; Park et al., 2021). In particular, this is the first theoretical account, to our knowledge, for the distance coding and hexagonal modulation characteristics in the human hippocampal formation (Park et al., 2021).

2 Background: Relational Inference Tasks in Human Experiments

To motivate our specific model design, we briefly review the tasks used in previous human experimental studies on relational learning. In Kumaran et al. (2012), the task is to learn a "one-dimensional hierarchy" or sequential ordering among 7 images of human faces or objects (Figure 1a). The true relations are not made known to the subject. During the learning phase, in each trial, the subject is presented with a pair of randomly chosen images that are adjacent in the ordering and required to infer the relation between them, either higher or lower (decision-making); the subject gets rewarded if the answer is correct (Figure 2a). After a certain number of trials, the subject's acquisition of the relation is assessed in a transitive inference task. This requires the subject to iteratively use the learned relation to relate a given pair of non-adjacent images. The study reported nearly perfect score for this task by the human subjects (Kumaran et al., 2012).

In Park et al. (2021), the task is to learn a "two-dimensional hierarchy" among 16 images of human faces (Figure 1b). The images are assumed to be organized in a 4×4 grid, which gives two types of sequential ordering among the individuals corresponding to the two axes ("competence and popularity"). The true relations are again unknown to the subject. The task proceeds similarly to the one-dimensional case, except that, in each trial, the subject is hinted with one of the two axes in which to infer the relation between a given pair of images (multi-axis). Also, unlike the one-dimensional case, each image can be related with up to four other images (many-to-many relation). After the learning procedure, the study conducted an fMRI experiment, which revealed distance coding and hexagonal modulation properties in the hippocampus and entorhinal cortex (Park et al., 2021).

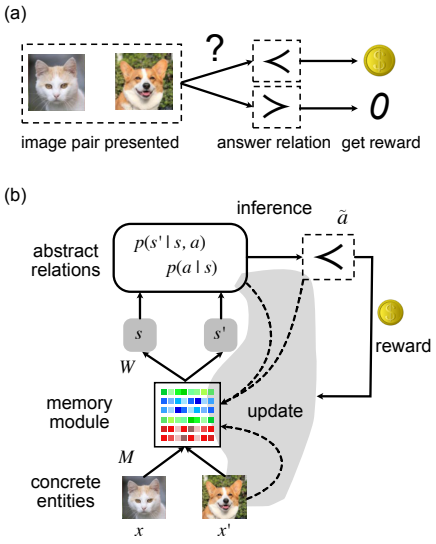


Figure 2: (a) Task design (each trial). (b) Model architecture.

procedure LEARN($\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_D, N$)
  initialize $\Phi = \{R_a, \sigma, \phi(g_a) | a \in A\} \cup \{W\}$
  for all $n = 1, \ldots, N$ **do**
    randomly select $d \in [1, D]$
    initialize M_1
    for all $t = 1, \ldots, T$ **do**
      observe (x_t, x'_t) in $\mathcal{R}_{\mathcal{D}_d}$
      $s_t = \text{infer}_{M_t, W}(x_t)$
      $s'_t = \text{infer}_{M_t, W}(x'_t)$
      sample $\tilde{a}_t \sim p(a_t | s_t, s'_t)$
      $r_t \leftarrow \text{get\_reward}((x_t, \tilde{a}_t, x'_t), \mathcal{R}_{\mathcal{D}_d})$
      if $r_t = 1$ **then**
        sample $\tilde{s}'_t \sim p(s'_t | s_t, \tilde{a}_t)$
        $\tilde{x}_t = \text{read}_{M_t, W}(s_t)$
        $\tilde{x}'_t = \text{read}_{M_t, W}(\tilde{s}'_t)$
        $M_{t+1} \leftarrow M_t +$
          $\alpha(x_t - \tilde{x}_t)\text{softmax}(W s_t)^\intercal +$
          $\alpha(x'_t - \tilde{x}'_t)\text{softmax}(W \tilde{s}'_t)^\intercal$
      $\mathcal{L} = \sum_{t=1}^{T} \left(r_t - p(\tilde{a}_t | s_t, s'_t)\right)^2$
        $+\gamma \left(r_t - p(\tilde{a}_t | s_t)\right)^2$
      miminize $\mathcal{L}$ w.r.t. Φ

Figure 3: Learning algorithm

3 FRAMEWORK

3.1 TASK FORMALIZED

Inspired by the human experiments overviewed in Section 2, we formalize the task design used for training and evaluation of our model. First, we assume a *relationship* $\mathcal{A}$ consisting of a finite set of relations. We also assume a *domain* $\mathcal{D} = (\mathcal{E}_\mathcal{D}, \mathcal{R}_\mathcal{D})$ with a finite set $\mathcal{E}_\mathcal{D} \subset \mathbb{R}^E$ of concrete entities or simply *entities* and a set $\mathcal{R}_\mathcal{D}$ consisting of tuples (x_i, a_i, x'_i) with $x_i, x'_i \in \mathcal{E}_\mathcal{D}$ and $a_i \in \mathcal{A}$. In the sequel, we generally consider a set of domains, in all of which the relations impose the same structure among the entities. As an example, consider the "ordering" relationship with two relations $\prec$ (prior to) and $\succ$ (next to), as in Figure 1a, which has the following general structure. The two relations are converse to each other: if a domain has $(x_1, \prec, x_2)$, then it also has $(x_2, \succ, x_1)$, and vice versa; in the sequel, we sometimes write $x_1 \prec x_2$ instead of $(x_1, \prec, x_2)$. In addition, every domain has a "minimal" entity $x_\perp$ for which there is no x' such that $(x', \prec, x_\perp) \in \mathcal{R}_\mathcal{D}$; similarly, every domain has a "maximal" entity $x_\top$; we call these *terminal* entities.

Our goal is to learn the general relational structure hidden in a given set of domains by performing the following relational inference task. The entire task undergoes a series of epochs. In each epoch, given some domain $\mathcal{D}$ (unknown to the model), the following process is repeated for T times: at time t, (i) receive a (random adjacent) pair of entities x_t, x'_t, (ii) infer their relation a_t, and (iii) obtain an immediate reward $r_t = 1$ if the inference is correct, i.e., $(x_t, a_t, x'_t) \in \mathcal{R}_\mathcal{D}$, or no reward $r_t = 0$ otherwise (Figure 2a). In the training phase, our model learns relational representation by performing the above process while maximizing total rewards for training domains. In the test phase, the model performs similar process for held-out test domains, which have new entities but with the same relational structure; the total rewards give the performance score. Note that it is the test phase that corresponds to the human tasks described in Section 2; the training phase would correspond to the subject's all experience prior to the experiments. Our model adopts the same task design for both phases for simplicity. Note also that the task structure is different from random-walking (Whittington et al., 2018; 2020), where we receive an entity x_t and a relation a_t in each step and predict the next entity x_{t+1} such that $(x_t, a_t, x_{t+1}) \in \mathcal{R}_\mathcal{D}$.

3.2 MODEL STRUCTURE

Our model has an architecture with two modules, one representing abstract relations and the other representing concrete entities (Figure 2b). Although such modular architecture is in common with

TEM (Whittington et al., 2018; 2020) and somewhat with other models (Graves et al., 2014; 2016; Webb et al., 2020), our main novelty lies in the learning algorithm described in Section 3.3.

To represent abstract relations, we assume abstract entities or *states* $s \in \mathbb{R}^S$. We define the "relational" probability $p(s'|s, a)$ that state s' is related with given state s by given relation $a \in \mathcal{A}$:

$$p(s'|s, a) = \mathcal{N}(\rho(R_a s), \sigma^2 I) \tag{1}$$

where $R_a \in \mathbb{R}^{S \times S}$ is a *relation matrix* specific to relation a and σ is a (global) scalar; ρ is an activation function, for which we use the L_2-normalization $\rho(s) = s/\|s\|$. We also define the "prior" probability $p(a|s)$ that given state s has relation a (with some other state):

$$p(a|s) = g_a(s) \quad \text{where} \sum_a g_a(s) = 1 \tag{2}$$

where $g_a(s)$ is a non-linear function specific to a. The prior distribution is particularly important to represent terminal entities, e.g., $p(\prec |s) \approx 0$ for maximal entities.

To represent concrete entities and their association with states, we introduce a memory mechanism. Assuming a *key matrix* $W \in \mathbb{R}^{H \times S}$ and a *memory matrix* $M \in \mathbb{R}^{E \times H}$, we refer to an entity corresponding to a state s by the following function:

$$\mathrm{read}_{M,W}(s) = Mh \quad \text{where } h = \mathrm{softmax}(Ws) \tag{3}$$

Here, the intermediate variable h softly represents an address pointing to the content of an entity stored in the memory. As discussed below, we take the key matrix W as a parameter, but the memory matrix M as a hidden variable. The latter allows for dynamically binding concrete entities with abstract states, which is crucial for discovering domain-general relational structure.

3.3 Learning algorithm

Given a set $\mathcal{D}_1, \ldots, \mathcal{D}_D$ of domains, our learning procedure runs a series of epochs described as follows (Figure 2b). In each epoch, we start with randomly selecting a domain $\mathcal{D}_d$ and randomly initializing the memory matrix M_1. At each time t, given a pair of entities x_t, x'_t, we first obtain the corresponding states $s_t = \mathrm{infer}_{M_t,W}(x_t)$ and $s'_t = \mathrm{infer}_{M_t,W}(x'_t)$ using the following "content-based" inference function:

$$\mathrm{infer}_{M,W}(x) = W^{\mathsf{T}} h \quad \text{where } h = \mathrm{softmax}(M^{\mathsf{T}} x) \tag{4}$$

After this, we make decision on the relation between these by sampling:

$$\tilde{a}_t \sim p(a_t|s_t, s'_t) \tag{5}$$

where the "posterior" distribution $p(a_t|s_t, s'_t)$ can be obtained by Bayes' rule using equations 1 and 2. The reward given for the decision is $r_t = 1$ if $(x_t, \tilde{a}_t, x'_t) \in \mathcal{R}_{\mathcal{D}_d}$, or $r_t = 0$ otherwise. Note that we use the memory to estimate both states (s_t and s'_t) and the relational representation to infer their relation (a_t), rather than using the relational representation to estimate the next state (s_t) from given previous state (s_{t-1}) and relation (a_t) as in an ordinary recurrent network or TEM (Whittington et al., 2018; 2020).

In the rewarded case, we adjust the memory to reflect the inference result. For this, we first re-postulate that the second state comes from the distribution for the inferred relation:

$$\tilde{s'_t} \sim p(s'_t|s_t, \tilde{a}_t) \tag{6}$$

(but leave the first state as it is for simplicity). We then recall the current memory contents $\tilde{x_t} = \mathrm{read}_{M_t,W}(s_t)$ and $\tilde{x'_t} = \mathrm{read}_{M_t,W}(\tilde{s'_t})$ and simultaneously update the memory:

$$M_{t+1} \leftarrow M_t + \alpha \left[(x_t - \tilde{x_t})\mathrm{softmax}(Ws_t)^{\mathsf{T}} + (x'_t - \tilde{x'_t})\mathrm{softmax}(W\tilde{s'_t})^{\mathsf{T}} \right] \tag{7}$$

In the update, we use a relatively large coefficient, e.g., $\alpha = 0.7$, which results in a drastic change of the memory contents from the old ones ($\tilde{x_t}$ and $\tilde{x'_t}$) to the new ones (x_t and x'_t). This update is a crucial step to make connection between concrete entities and abstract relation. Note that our memory mechanism is rather different from Hopfield-type auto-associative memory like Ba et al. (2016), which is used in TEM (Whittington et al., 2018; 2020), where the update rule only stores new

associations, not erasing old ones. In this sense, our memory is more similar to Graves et al. (2014; 2016), although the old content is estimated directly from the memory in our update rule (equation 7), whereas it is computed by the controller network in their case. As shown in the experiment in Section 4.1, the choice of memory mechanism is essential in our setting. This is probably because our task induces crucial interaction between states and memory and thus earlier inaccurate state-entity associations must be replaced later. (Such problem does not seem to arise in TEM since current states can be inferred mainly from previous states.)

Then, we define the following one-step loss function, which encourages correct inferences and discourage incorrect ones:

$$\mathcal{L}_t = (r_t - p(\tilde{a_t}|s_t, s'_t))^2 \tag{8}$$

Our goal is to minimize this over steps and epochs with respect to the parameters $\Phi = \{R_a, \phi(g_a)|a \in A\} \cup \{W, \sigma\}$, where $\phi(\cdot)$ is the set of parameters used in the given function. Note that the loss function depends on the states, which in turn depend on the memory, which further depends on the previous memory, inputs, etc. Therefore optimizing the loss necessarily causes back-propagation through time.[2] We stress that recurrent computation here occurs primarily through memory rather than through states, which is somewhat unique to our model.

Lastly, we incorporate a simple regularization that enforces the prior to follow the actual occurrences of the relations.

$$\mathcal{L}_t^{\texttt{prior}} = (r_t - p(\tilde{a_t}|s_t))^2 \tag{9}$$

In our experience, without this regularization, the learned prior tends to be uniform. This makes the model fail to capture terminal entities and thereby degrade performance in transitive inference (Section 4.1). The entire learning procedure is summarized as pseudo-code in Figure 3.

3.4 TRANSITIVE INFERENCE

Transitive inference is an excellent task to test the learned relational representation. In this, given a pair x, x' of entities, we ask to infer the relation a whose one or more iterate, or *transitive closure*, relates them: $(x, a, x_1), (x_1, a, x_2), \ldots, (x_{m-1}, a, x) \in \mathcal{D}$ for some $m \geq 1$ (where the subscript is for iteration, not for time).

To solve this, we first obtain the states s, s' corresponding to x, x'. We then calculate the following two probabilities $p(s'|s, a^m, m)$ (that state s' is related with given state s by the m-th iterate of given relation a) and $p(a^m|s, m)$ (that given state s has m-th iterate of relation a):

$$p(s'|s, a^m, m) = p(s'|\psi_a^{m-1}(s), a) \qquad p(a^m|s, m) = \prod_{i=0}^{m-1} p(a|\psi_a^i(s)) \tag{10}$$

where $\psi_a(s) = \kappa(\rho(R_a s))$ and $\kappa(s) = \mathrm{infer}_{M,W}(\mathrm{read}_{M,W}(s))$. Using these, we next obtain the following probability $p^+(a|s, s')$ (that given states s and s' are related by the transitive closure of relation a):

$$p^+(a|s, s') = \frac{1}{M} \sum_{m=1}^{M} \frac{p(s'|s, a^m, m)p(a^m|s, m)}{\sum_{a'} p(s'|s, a'^m, m)p(a'^m|s, m)} \tag{11}$$

(assuming that only the same relation is iterated at most M times). Finally, we answer $a^+ = \arg\max_a p^+(a|s, s')$ for the relation in question. For later evaluation, we also use the confidence value $c^+ = \max_a p^+(a|s, s')$.

The above approach can actually be derived as an approximation in a certain probabilistic framework (Section A). The approach works in the case of one-to-one relations as used in our experiment in Section 4.1. Note that, for successful transitive inference, it is crucial for the model to learn both distributions $p(s'|s, a)$ and $p(a|s)$ precisely; otherwise, we may wrongly judge, e.g., that a state has the m-th iterate of a even if it does not.

[2]Note that the learning procedure involves discrete sampling (equation 5), which prevents some gradients from being propagated. Empirically, however, our algorithm stably optimizes (Section 4), possibly using imprecisely computed gradients. We observed no notable improvement when using a straight-through estimator (Bengio et al., 2013; Jang et al., 2017).

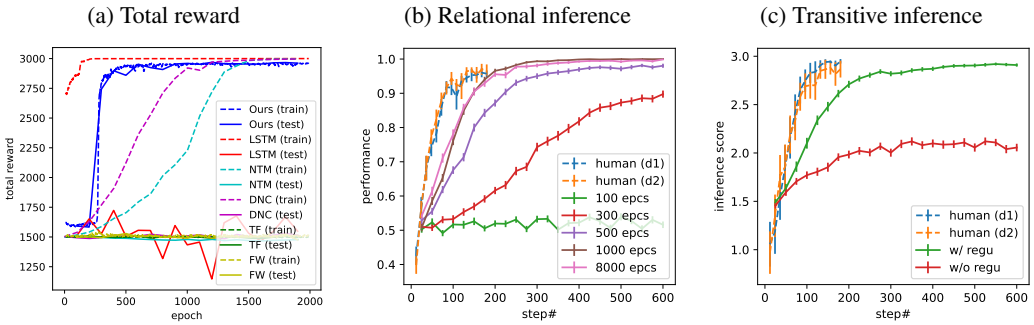


Figure 4: Results on the one-dimensional hierarchy task. (a) Traces (up to 2000 epochs) of total rewards for our model for training and test data, contrasted with LSTM, NTM, DNC, Transformer (TF), and Fast Weights (FW). For NTM and DNC, which took very long training time (about 40 days), the horizontal axis is 50-times reduced for readability. (b) Within-epoch performance ramp (up to 600 steps) in the relational inference at different training stages and the corresponding human data (for two domains). (c) Inference score in the transitive inference for the models trained with and without regularization, together with human data. The human data are replotted from Kumaran et al. (2012). Error bars show standard errors over domains (models) or subjects (humans).

4 Experiments

In this section, we present the results of experiments on two tasks. Additional details on the experiments are given in Sections B and C.

4.1 One-dimensional hierarchy

The first experiment here simulates the previous human experiment on a simple one-dimensional hierarchy (Kumaran et al., 2012) outlined in Section 2. Using CIFAR100, we formed each domain consisting of 7 images as entities, randomly selected and ordered from a single, randomly chosen object class. We prepared 600 training domains and 100 test domains so that these included images from disjoint classes. We assumed two relations $\prec$ and $\succ$, which intendedly meant that $x \prec x'$ if x is immediately prior to x' and that $x \succ x'$ if x is immediately next to x' (Figure 1a). Note that these relations are defined only for adjacent pairs; there are a minimal and a maximal entities. Note also the small-size nature of the task: only 12 possible pairs per domain. In the training phase, starting with a randomly initialized model ($H = 50$; $S = 20$), we ran 8000 epochs each performing 3000 training steps for a randomly chosen domain. The training took about 3 days. We then proceeded to the test phase, which was similar to training, but without optimization with respect to the model parameters. Below, we present the results.

First, Figure 4a shows the trace of per-epoch total rewards during training and test for a model instance. The training succeeded with almost full rewards. Importantly, the test followed a similar trace to the training, indicating that the model successfully generalized the relational structure to unseen entities. Figure 4a also gives comparison with other existing models trained on the same task (see Section B.2.2 for details). (1) LSTM (Hochreiter and Schmidhuber, 1997), a conventional recurrent model, showed no such generalization, which is no surprise since it has no updatable memory. (2) NTM (Graves et al., 2014) and NDC (Graves et al., 2016), recurrent models with memory mechanism, also showed no generalization, despite their complex and powerful capability. This can be because that these models do not decouple well abstract and concrete representations (e.g., values to write in to memory are determined only by the recurrent network). (3) vanilla Transformer (Vaswani et al., 2017) did not succeed even in training; we tried various architecture and training settings in vain. We consider that the model structure does not match our task nature with very small dataset and high-dimensional input so that optimization easily falls into poor local minima.[3] (4) A modified version of our model using Fast Weights, a Hopfield-type memory mechanism (Ba et al.,

[3]Whittington et al. (2022) shows that Transformer can accommodate a version of TEM, for which no such optimization problem happens probably because they restrict weight matrices in a particular way.

2016), also failed training. The model can be seen as a version of TEM (Whittington et al., 2018; 2020) replacing the learning algorithm with ours (to match the task format), but retaining the memory mechanism. The shown result thus implies that the appropriate choice of memory mechanism highly depends on the task structure. In sum, our comparisons above highlight the particular difficulty of our task that leads these powerful models to failure.

Second, we inspected how the model behaves within an epoch on test data. Figure 4b shows the trace of per-block test performance (probability of reward in a block of 25 steps) in the relational inference within an epoch at different stages of training. In each epoch, the model started to perform badly but gradually became better. This behavior is expected since the model initially knows nothing about new entities but gets acquainted with their association with the learned relation during the epoch. Indeed, as the representation got improved during training, the time to gain full rewards (thus identify the correct relations) became shorter and shorter. After completion of training (8000 epochs), the peak performance is comparable with the human data (Kumaran et al., 2012) replotted in Figure 4b. (We are not concerned here about the speed of ramp as human brains can clearly do much more complex operations in a single trial, such as replays and some kind of inferences—the model can at best simulate human at the abstract level.[4])

Third, we tested how well the model could exploit the learned representation for transitive inference task, using the scheme described in Section 3.4. To show the result in comparison to human data, we calculated the inference score, namely, the performance multiplied by the confidence value ($\lfloor 2.99 c^{\mathtt{ti}} + 1 \rfloor$). Figure 4c shows that the inference score ramped in accordance with the relational inference in Figure 4b. Again, the peak score is comparable to the corresponding human data (Kumaran et al., 2012), replotted in Figure 4c. In addition, Figure 4c shows that, when a model was trained without the regularization described in Section 3.3 and therefore did not precisely learn the prior distribution, the performance became significantly poorer. This result suggests that, in the human experiment, although the task did not explicitly require to infer the terminalities of entities, humans might have recognized these implicitly and thereby achieved the high task score. This result can stand as a testable prediction.

Finally, we trained 8 instances of models on the same task with the same condition and measured the variance of the performance. Table 1 summarizes the results, confirming their robustness.

4.2 TWO-DIMENSIONAL HIERARCHY

The second experiment simulates the previous human experiment on the two-dimensional hierarchy task (Park et al., 2021), as outlined in Section 2. Similarly to the 1D case, we formed training and test domains each consisting of 16 images as entities. As noted, the task involve multi-axis, many-to-many relations. That is, the 16 entities intendedly formed 4×4 grid, with ordering in two axes: $\prec_1$ and $\succ_1$ in axis 1, and $\prec_2$ and $\succ_2$ in axis 2 (Figure 1b). Also, in one axis, each entity can be related with up to four entities; e.g., $x \prec_1 x'$ holds for entities that are adjacent in axis 1, but potentially non-adjacent in axis 2.

We trained a model ($H = 500, S = 20$) for 6000 epochs, with 2000 steps per epoch. In each epoch, a target axis i was randomly chosen and given to the model so that it could infer between $\prec_i$ and $\succ_i$. The training conditions were otherwise similar to Section 4.1. For comparison, we trained another model with smaller memory ($H = 50$) and a null model (trained with the same condition but random rewarding).[5] The training of each model took about 2 days.

As a basic assessment, we ran the two trained models in the relational inference task in test domains and confirmed performance ramp to a reasonably high score for both models (Figure 5a). Our main interest here is, however, on how the learned intermediate (h) and state (s) representations compared to the human hippocampus and entorhinal cortex, respectively, in a similar task. For this, we performed the following analyses on the model after running it in each test domain.

[4]Note that both the model and human are much slower than the ideal observer (Section B.2.2), which reaches nearly 1.0 around 50 steps in our calculation.

[5]Since no other model so far has succeeded in solving our task and, we believe, neural plausibility of a non-working model is meaningless, we provide here no answer as to whether or not the similarity with the human fMRI data is unique to our model. In future studies, however, when other models turn out to be able to solve the task, the same question should be revisited and investigated.

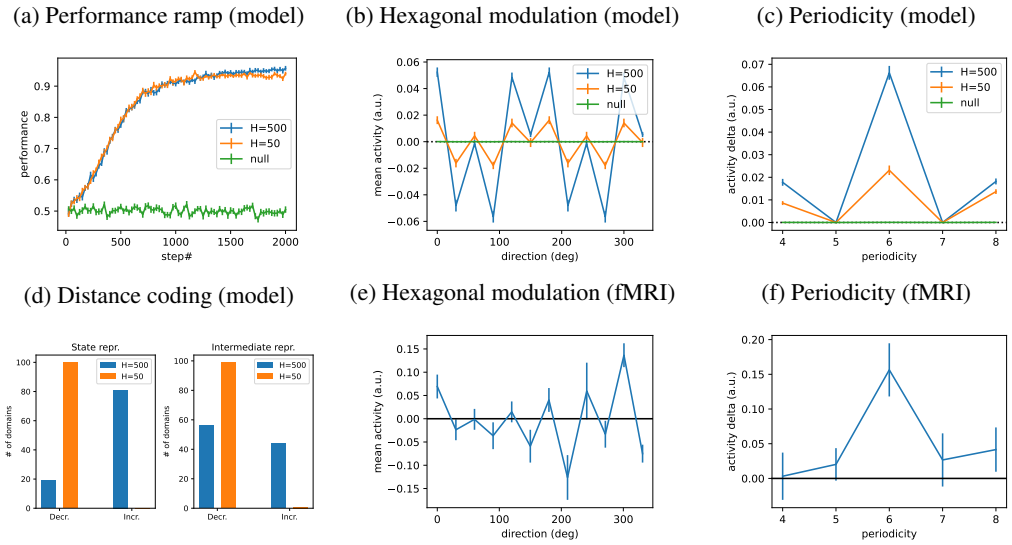


Figure 5: Results on the two-dimensional hierarchy task. (a) Performance ramp in the relational inference. (b, c) The hexagonal modulation property and estimated strength at each periodicity. (d) Distance coding in the state (left) or intermediate (right) representations. The results for models with larger ($H = 500$) or smaller memory ($H = 50$) as well as a null model are shown. (e, f) Analogous results to (b, c) in the fMRI experiment replotted from Park et al. (2021) (Fig. 4b). Error bars show standard errors over domains (models) or participants (humans).

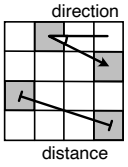


Figure 6: Grid coding

First, Park et al. (2021) reported that fMRI signals from the entorhinal cortex during presentation of two consecutive stimuli showed hexagonal modulation with respect to the grid direction (Figure 6). To analyze our state representation in a compatible manner (Appendix C.1), we computed the (z-scored) state vectors corresponding to the images at each pair of grid positions and simply took their average. The state values, when plotted against the direction (angle) between the two positions, overall showed a periodic pattern of increase and decrease in the cycle of $60\,\mathrm{deg}$; Figure 5b shows the phase-aligned mean within each bin of $30\,\mathrm{deg}$. To quantify the periodicity, we calculated the mean differences between the state values at multiples of $60\,\mathrm{deg}$ and the rest (6-fold periodicity) and repeated this for $90\,\mathrm{deg}$, $72\,\mathrm{deg}$, $51.4\,\mathrm{deg}$, and $45\,\mathrm{deg}$ (4-, 5-, 7-, and 8-fold periodicities, respectively); the 6-fold periodicity gave the maximal difference (Figure 5c). The results were overall similar between the two trained models (but with lower magnitudes for the smaller memory), resembling the human fMRI data (Park et al., 2021) (replotted in Figure 5ef).

Second, Park et al. (2021) observed that signals from the hippocampus and entorhinal cortex showed increasing patterns for distance coding (Figure 6). To analyze our state and intermediate representations in a compatible manner to the experiment, we computed the dissimilarity (Euclidean distance) between the vectors corresponding to the images at each pair of grid positions (Appendix C.1). Repeating these for different domains, the dissimilarity had either increasing or decreasing tendencies against the (Euclidean) distance in the grid space. (In other words, increasing tendency indicates that the representations are more similar between closer grid positions.) Figure 5d summarizes the number of domains with increase or decrease in the intermediate or state representation. In both representations, increasing tendencies were rarely found in the model with smaller memory, but much more often with larger memory, thus more similar to the human fMRI data reporting only increasing tendencies (Park et al., 2021). These results are robust across model instances trained in the same condition (Appendix C.2).

The hexagonal modulation in the human entorhinal cortex is closely related to the grid cell property found in the rodent entorhinal cortex (Hafting et al., 2005), since such modulation is expected as mixed neural signals from grid cells if they have hexagonal grid fields as in the rodent. In this sense,

our result reproducing the hexagonal modulation is related to previous model studies reproducing the grid cell property such as Whittington et al. (2018; 2020). However, we stress that neither result is directly predictable from the other since the task and model structures are rather different.

5 Related work

Recently, learning models for the cognitive map representations in the hippocampal formation have been drawing attention. Among others, Whittington et al. (2018; 2020) proposed TEM, which accounts for place and grid cells in rodents as an abstract relational structure of 2D-geometric environments while performing spatial random-walking tasks. Our study has been much influenced by theirs, adopting a similar modular architecture with separate relation and memory representations. However, the different task goal, decision-making, necessitates our model construction to use different approaches in (1) the learning algorithm that maximizes rewards from memory-based relational inference and (2) the memory mechanism with the update rule that can replace old contents. In particular, we showed that the Hopfield-type memory proposed by Ba et al. (2016) does not work in our setting (Section 4.1), which is quite striking, given that TEM was successful by using this mechanism: a simplistic adaption of TEM to our task fails. Later, Whittington et al. (2022) simplified TEM and clarified the relationship with Transformers (Vaswani et al., 2017). In this, they introduced another memory mechanism with a query-key-value-type read operation, but their update rule only adds new associations without erasing old ones (their Appendix). The modular architecture for learning to represent the general 2D structure has also been proposed by Uria et al. (2022), which also explained the variety of hippocampal cells. In different approaches, place and grid cell properties have also been reproduced in intermediate representations in recurrent networks (Banino et al., 2018; Sorscher et al., 2019) or successor representations in reinforcement learning (Stachenfeld et al., 2017).

Memory-augmented recurrent models have been used for solving general, complex tasks requiring relational reasoning. Some such models use memory for connecting abstract relations with concrete entities to discover abstract rules in input sequences and thereby solve symbol-processing tasks (Webb et al., 2020; Chen et al., 2021). However, despite the apparent similarity, their tasks to find out common rules hidden within fixed-length sequences are not compatible with our tasks to find out general structure in a set of binary relations. Graves et al. (2014; 2016) have proposed recurrent networks with external memory, highly influenced by von Neumann machines, that can learn to solve list and graph problems. Although these models are powerful, their way of coupling the recurrent network and the memory module seems to deteriorate disentangled representation of abstract relations and concrete domains (Section 4.1). Santoro et al. (2018) has presented a different approach for relational reasoning by using multi-head attention that allows for interaction between memory slots.

Recent studies have incorporated memory mechanisms in classifier neural network (Ba et al., 2016), in reinforcement learning (Pritzel et al., 2017; Hansen et al., 2018; Fortunato et al., 2019), in generative models (Li et al., 2016; Bornschein et al., 2017; Wu et al., 2018), in meta-learning (Santoro et al., 2016; Munkhdalai and Yu, 2017), and so on. Although these models have some technical commonalities with ours, they use memory to efficiently recall past experience and thereby increase specific task performance; therefore the goals are largely different from ours.

6 Conclusion

To pursue the recently emerging learning principle of abstract relational structure hypothesized for the hippocampal formation, we have developed a novel memory-based cognitive model for solving decision-making tasks and compared it with previous human behavioral and fMRI data. The results showing a good match suggest that this principle can potentially be extended from previously shown spatial random-walking to more general relational learning tasks. Future studies should further investigate neural representations of other relational structures like trees and cycles, interaction with other brain functions like the prefrontal cortex, and connection with neural data in rodents.

REFERENCES

Jimmy Ba, Geoffrey Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. *Advances in Neural Information Processing Systems*, pages 4338–4346, 2016.

Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, Greg Wayne, Hubert Soyer, Fabio Viola, Brian Zhang, Ross Goroshin, Neil Rabinowitz, Razvan Pascanu, Charlie Beattie, Stig Petersen, Amir Sadik, Stephen Gaffney, Helen King, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, and Dharshan Kumaran. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, may 2018.

Xiaojun Bao, Eva Gjorgieva, Laura K. Shanahan, James D. Howard, Thorsten Kahnt, and Jay A. Gottfried. Grid-like Neural Representations Support Olfactory Navigation of a Two-Dimensional Odor Space. *Neuron*, 102(5):1066–1075.e5, 2019.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. pages 1–12, 2013.

Jörg Bornschein, Andriy Mnih, Daniel Zoran, and Danilo J. Rezende. Variational memory addressing in generative models. *Advances in Neural Information Processing Systems*, 2017-Decem:3921–3930, 2017.

M. Bunsey and H. Eichenbaum. Conservation of hippocampal memory function in rats and humans. *Nature*, 379(6562):255–257, jan 1996.

Rishidev Chaudhuri and Ila Fiete. Computational principles of memory. *Nature Neuroscience*, 19(3): 394–403, 2016.

Catherine Chen, Qihong Lu, Andre Beukers, Christopher Baldassano, and Kenneth A. Norman. Learning to perform role-filler binding with schematic knowledge. *PeerJ*, 9:1–27, 2021.

Alexandra O Constantinescu, Jill X. O'Reilly, and Timothy E.J. Behrens. Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468, 2016.

Jeffery A. Dusek and Howard Eichenbaum. The hippocampus and memory for orderly stimulus relations. *Proceedings of the National Academy of Sciences of the United States of America*, 94 (13):7109–7114, 1997.

Howard Eichenbaum and Neal J. Cohen. Can We Reconcile the Declarative Memory and Spatial Navigation Views on Hippocampal Function? *Neuron*, 83(4):764–770, 2014.

Moire Meire Fortunato, Melissa Tan, Ryan Faulkner, Steven Sleven Hansen, Adrift Puifidomenech Badla, Gavin Buttlmore, Charles Charlie Deck, Joel Z. Lelbo, Charles Blundell, Adrià Puigdomènech Badia, Gavin Buttimore, Charles Charlie Deck, Joel Z Leibo, and Charles Blundell. Generalization of Reinforcement Learners with Working and Episodic Memory. In H Wallach, H Larochelle, A Beygelzimer, F d'Alché-Buc, E Fox, and R Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 1–10, 2019.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing Machines. pages 1–26, oct 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626): 471–476, 2016.

Torkel Hafting, Marianne Fyhn, Sturla Molden, May Britt Moser, and Edvard I. Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806, 2005.

Steven S. Hansen, Pablo Sprechmann, Alexander Pritzel, André Barreto, and Charles Blundell. Fast deep reinforcement learning using online adjustments from the past. *Advances in Neural Information Processing Systems*, 2018-Decem(NeurIPS):10567–10577, 2018.

S Hochreiter and J Schmidhuber. Long short-term memory. *Neural Computation*, jan 1997.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–13, 2017.

D Kingma and J Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, jan 2015.

Dharshan Kumaran, Hans Ludwig Melo, and Emrah Duzel. The Emergence and Representation of Knowledge about Social and Nonsocial Hierarchies. *Neuron*, 76(3):653–666, 2012.

Chongxuan Li, Jun Zhu, and Bo Zhang. Learning to generate with memory. *33rd International Conference on Machine Learning, ICML 2016*, 3:1811–1822, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, apr 2016.

Edvard I. Moser, Emilio Kropff, and May Britt Moser. Place cells, grid cells, and the brain's spatial representation system. *Annual Review of Neuroscience*, 31:69–89, 2008.

Tsendsuren Munkhdalai and Hong Yu. Meta Networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2554–2563, International Convention Centre, Sydney, Australia, 2017.

Seongmin A. Park, Douglas S. Miller, Hamed Nili, Charan Ranganath, and Erie D. Boorman. Map Making: Constructing, Combining, and Inferring on Abstract Cognitive Maps. *Neuron*, 107(6): 1226–1238.e8, 2020.

Seongmin A. Park, Douglas S. Miller, and Erie D. Boorman. Inferences on a multidimensional social hierarchy use a grid-like code. *Nature Neuroscience*, 24(9):1292–1301, 2021.

Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural Episodic Control. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2827–2836, International Convention Centre, Sydney, Australia, 2017.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-Learning with Memory-Augmented Neural Networks. *33rd International Conference on Machine Learning, ICML 2016*, 4:2740–2751, 2016.

Adam Santoro, Ryan Faulkner, David Raposo, Jack Rae, Mike Chrzanowski, Théophane Weber, Daan Wierstra, Oriol Vinyals, Razvan Pascanu, and Timothy Lillicrap. Relational recurrent neural networks. *Advances in Neural Information Processing Systems*, 2018-Decem:7299–7310, 2018.

Ben Sorscher, Gabriel C. Mel, Surya Ganguli, and Samuel A. Ocko. A unified theory for the origin of grid cells through the lens of pattern formation. *Advances in Neural Information Processing Systems*, 32(NeurIPS):1–11, 2019.

Kimberly L Stachenfeld, Matthew M Botvinick, and Samuel J Gershman. The hippocampus as a predictive map. *Nature Neuroscience*, 20(11):1643–1653, 2017.

Benigno Uria, Borja Ibarz, Andrea Banino, Vinicius Zambaldi, Dharshan Kumaran, Demis Hassabis, Caswell Barry, and Charles Blundell. A model of egocentric to allocentric understanding in mammalian brains. *bioRxiv*, page 2020.11.11.378141, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. 2017.

Taylor W. Webb, Ishan Sinha, and Jonathan D. Cohen. Emergent Symbols through Binding in External Memory. pages 1–28, 2020.

James C.R. Whittington, Timothy H. Muller, Caswell Barry, Shirley Mark, and Timothy E.J. Behrens. Generalisation of structural knowledge in the hippocampal-entorhinal system. *Advances in Neural Information Processing Systems*, 2018-Decem(Nips):8484–8495, 2018.

James C.R. Whittington, Timothy H. Muller, Shirley Mark, Guifen Chen, Caswell Barry, Neil Burgess, and Timothy E.J. Behrens. The Tolman-Eichenbaum Machine: Unifying Space and Relational Memory through Generalization in the Hippocampal Formation. *Cell*, 183(5):1249–1263.e23, 2020.

James C.R. Whittington, Joseph Warren, and Timothy E.J. Behrens. Relating Transformers To Models and Neural Representations of the Hippocampal Formation. *ICLR 2022 - 10th International Conference on Learning Representations*, pages 1–20, 2022.

Yan Wu, Greg Wayne, Alex Graves, and Timothy Lillicrap. The kanerva machine: A generative distributed memory. *arXiv*, pages 1–16, 2018.

A Derivation of the approximation in Section 3.4

For solving the transitive inference task, we consider the probability $p(s'|s, a^m, m)$ that state s' is related with given state s by the m-th iterate of given relation a:

$$p(s'|s, a^m, m) = \iint \cdots \int \left[\prod_{i=0}^{m-2} p(s_{i+1}|s_i, a) \right] p(s'|s_{m-1}, a) ds_1 ds_2 \dots ds_{m-1} \tag{12}$$

and the probability $p(a^m|s, m)$ that given state s has m-th iterate of relation a:

$$p(a^m|s, m) = \iint \cdots \int \left[\prod_{i=0}^{m-2} p(s_{i+1}, a|s_i) \right] p(a|s_{m-1}) ds_1 ds_2 \dots ds_{m-1} \tag{13}$$

The above definitions are intractable in general. However, we use the following approximation assuming one-to-one relations:

$$p(s'|s, a^m, m) \approx p(s'|\psi_a^{m-1}(s), a) \qquad p(a^m|s, m) \approx \prod_{i=0}^{m-1} p(a|\psi_a^i(s)) \tag{14}$$

where $\psi_a(s) = \rho(R_a s)$. (Empirically, we find it more useful to "clean up" the state after each iteration, and therefore use $\psi_a(s) = \kappa(\rho(R_a s))$ with the function $\kappa(s) = \text{infer}_{M,W}(\text{read}_{M,W}(s))$, which recalls the concrete entity of a given state vector and then mapping it back to a state.)

To derive the above, we crudely approximate the Gaussian distribution in equation 1 by a delta function: $p(s'|s, a) \approx \delta[s' = \rho(R_a s)]$. To derive the approximation of $p(s'|s, a^m, m)$ in equation 14, first let:

$$A_{m,a}^k = \iint \cdots \int \left[\prod_{i=0}^{m-2-k} p(s_{i+1}|s_i, a) \right] p(s'|\psi_a^k(s_{m-1-k}), a) ds_1 ds_2 \dots ds_{m-1-k} \tag{15}$$

For $0 \leq k \leq m-2$, by the approximation assumption $p(s'|s, a) \approx \delta[s' = \psi_a(s)]$, we obtain:

$$\begin{aligned} A_{m,a}^k \approx \iint \cdots \int \left[\prod_{i=0}^{m-3-k} p(s_{i+1}|s_i, a) \right] \delta[s_{m-1-k} = \psi_a(s_{m-2-k})] \\ p(s'|\psi_a^k(s_{m-1-k}), a) ds_1 ds_2 \dots ds_{m-1-k} \end{aligned} \tag{16}$$

$$= \iint \cdots \int \left[\prod_{i=0}^{m-3-k} p(s_{i+1}|s_i, a) \right] p(s'|\psi_a^{k+1}(s_{m-2-k}), a) ds_1 ds_2 \dots ds_{m-2-k} \tag{17}$$

$$= A_{m,a}^{k+1} \tag{18}$$

Thus, noting $A^0_{m,a} = p(s'|s, a^m, m)$, the result follows.

To derive the approximation of $p(a^m|s, m)$ in equation 14, let:

$$B^k_{m,a} = \iint \cdots \int \left[\prod_{i=0}^{m-2-k} p(s_{i+1}, a|s_i) \right] \prod_{i=0}^{k} p(a|\psi^i_a(s_{m-1-k}))ds_1 ds_2 \dots ds_{m-1-k} \tag{19}$$

For $0 \leq k \leq m-2$, by the same approximation assumption, we obtain:

$$B^k_{m,a} \approx \iint \cdots \int \left[\prod_{i=0}^{m-3-k} p(s_{i+1}, a|s_i) \right] \delta[s_{m-1-k} = \psi_a(s_{m-2-k})] p(a|s_{m-2-k}) \left[\prod_{i=0}^{k} p(a|\psi^i_a(s_{m-1-k})) \right] ds_1 ds_2 \dots ds_{m-1-k} \tag{20}$$

$$= \iint \cdots \int \left[\prod_{i=0}^{m-3-k} p(s_{i+1}, a|s_i) \right] \left[\prod_{i=0}^{k+1} p(a|\psi^i_a(s_{m-2-k})) \right] ds_1 ds_2 \dots ds_{m-2-k} \tag{21}$$

$$= B^{k+1}_{m,a} \tag{22}$$

Noting $B^0_{m,a} = p(a^m|s, m)$, the result follows.

B Addendum on the experiment in Section 4.1

B.1 Datasets

In each experiment, we prepared a dataset using the CIFAR100 image database. To form each domain, we randomly chose an object class from which we randomly selected 7 images as entities and then ordered them randomly. We formed 600 training domains and 100 test domains, with no overlapping object classes between the two. Each image was compressed to 700 dimensions by PCA and normalized to unit norm.

B.2 Training

B.2.1 Our model

For the architecture, apart from the description in Section 3, we used a two-layer perceptron to implement $g_a(s)$ (for prior) where the intermediate layer had 10 units with the sigmoid activation function and the output layer was fed to the softmax function.

To train the model, we used hyper-parameters $\alpha = 0.7$ and $\gamma = 1$, and Adam optimizer (Kingma and Ba, 2015) with mini-batch size 5. We also used the same $\alpha = 0.7$ for test. To boost the learning, we used a truncated back-propagation strategy. More precisely, we ran optimization after every 25 steps in an epoch. That is, after every 25 steps, we first initialized the accumulated loss function and stopped gradient propagation for the memory matrix M.

Generally, from observation of the loss function, training tends to proceed as follows. Initially, learning starts with a lengthy plateau (where the loss scarcely changes), then suddenly shifts to a rapid drop (where the loss quickly improves), and thereafter falls into another plateau. This stairway-like course continues for a few times and finally reaches a very long and slow slope for convergence. In some cases, the initial plateau is so long that it is unclear whether the optimization simply fails, in which case we manually stop it after around 2000 epochs and start it over.

B.2.2 Baseline models

We adopted the following baseline models.

Long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) A well-known recurrent model . Number of layers: 1. Hidden state dimensions: 70.

Table 1: Summary of results of relational inference (performance) and transitive inference (inference score) on one-dimensional hierarchy (at 1000 steps in each epoch). The mean and s.d. of scores over 8 model instances are shown.

	Ours	Ours w/o reg.	LSTM	TF	FW	Human (d1)	Human (d2)
Relational Inf.	1.00 ± 0.00	1.00 ± 0.00	0.51 ± 0.02	0.50 ± 0.00	0.51 ± 0.01	0.96	0.97
Transitive Inf.	2.86 ± 0.07	2.09 ± 0.14	—	—	—	2.95	2.87

Neural Turing Machine (NTM) (Graves et al., 2014) A recurrent model with updatable external memory. Hidden state dimension: 20. Memory height (number of slots): 50. Memory width (content dimensions): 100

Differentiable Neural Computer (DNC) (Graves et al., 2016) An extension of NTM. Hidden state dimension: 20. Memory height: 50. Memory width: 100. Number of write heads: 1. Number of read heads: 4.

vanilla Transformer (Vaswani et al., 2017) A well-known language model incorporating self-attention with (additive) sine/cosine position-encoding and temporal masking. Number of layers: 2 (repeated). Number of heads: 1. Hidden state dimension: 1400. Drop-out rate: 0.1. Context size: 100.

Fast Weight A modified version of our model using the auto-associative memory proposed in Ba et al. (2016). The encoding/decoding method of state and content described in Whittington et al. (2020, Section 4) is used.

In all of these, we input the concatenation of two given images to the model at each step and the correct relation as target. For NTM and DNC, we also append the correct relation in one step to the input in the next step, so that the model can take the correctness of the previous inference into account. For Transformer, we give the concatenated image pair and the target alternately in the input and let the model infer the target from each image pair, whose summed squired error becomes the loss function.

In addition, we examined the ideal observer, which performs as well as possible in a given epoch, assuming that it knows perfectly the 1D task structure. We implemented the ideal observer such that it memorizes all past pairs in the epoch and, at each step, simply searches for the given pair or its flipped pair; if it finds, then it can give the right answer (reward 1) or otherwise a random answer (reward 0.5).

B.3 Additional results

Table 1 summarizes the results from 8 model instances for each task and for each model (except NTM and DNC for which we could train only one instance due to the very long training time), together with the human data. Figure 7 shows the traces of performance in relational inference and inference score in transitive inference for all models.

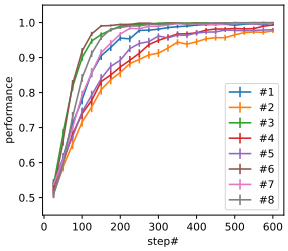
For Transformer, we tried a number of other architecture parameter settings than described in Section B.2.2, using more layers (5), more heads (2 or 5), lower input dimensions (2, 10, or, 100), or larger context sizes (300). (The hidden state dimensions were the same as the input dimensions.) The result was similar: all models completely failed even in training. This result was somewhat surprising to us since the task initially appeared to be easy for Transformer. Indeed, one can imagine a Transformer that uses the input directly as a query and key and then uses position encoding to find the corresponding relation found in the next entry in the sequence. However, optimization never found out such solution but seemed to just fall into meaningless local minima.

C Addendum on the experiment in Section 4.2

C.1 Analysis method

Our aim here is to analyze the internal representations in out trained model in a way comparable to the brain analysis described in Park et al. (2021).

(a) Relational inference



(b) Transitive inference

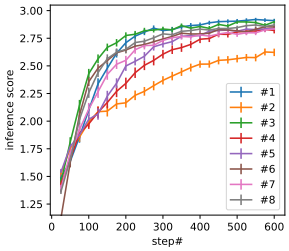


Figure 7: Additional results on one-dimensional hierarchy: (a) the traces of performance in relational inference and (b) inference score in transitive inference for all models.

First, in the fMRI experiment (Park et al., 2021), they analyzed the hexagonal modulation property of the brain signals from the entorhinal cortex during two consecutive visual stimuli corresponding to two grid positions. To simulate this, after running the model in a given test domain, we obtained the state vector corresponding to the image at each grid position by using inference through the memory and took the average over the two state vectors. The values in state values are normalized as z-scores. By plotting the state values in each dimension against the angle between the grid positions, we obtained a function over angles. Repeating this for different dimensions, we obtained a set of such functions. To see if those functions had periodicity in a given cycle ($90\,\mathrm{deg}$, $72\,\mathrm{deg}$, $60\,\mathrm{deg}$, $51.4\,\mathrm{deg}$, and $45\,\mathrm{deg}$), we visualize the average over the functions, where we needed to deal with that each function may have a different phase. In Park et al. (2021), to avoid contamination of phase and periodicity, they estimated the phases by using another dataset obtained from a separate experiment with the same task condition. We simulated this by using the state vectors obtained through the model at 25 steps prior to the completion of running in the test domain (which has a different memory matrix). To estimate the phase, following the method in Park et al. (2021), we fit state values in each dimension against the sin and cos of the angle and took the arc-tangent of the regression coefficients. We then calculated the circular mean over the obtained phases. Finally, we took the phase-aligned average of the functions for all dimensions separately for each domain, and then averaged the resulting functions for all domains.

Second, in the same experiment Park et al. (2021), they analyzed the distance coding in the brain signals from the hippocampus and entorhinal cortex. To simulate this, we similarly obtained the intermediate or state vectors corresponding to the images at each pair of grid positions and calculated the distance between these as their dissimilarity. Although (Park et al., 2021) used Maharanobis distance, we used Euclidean distance since our case uses the deterministic state inference. By plotting the dissimilarity against the Euclidean distance between the grid positions, we determined the increase or decrease tendency by the correlation coefficient (positive or negative).

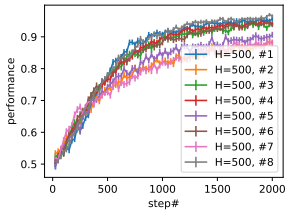
C.2 Additional results

We trained 4 models with larger ($H = 500$) or smaller ($H = 50$) memory. Figure 8 summarizes the results for these in the same format as Figure 5. These show overall robustness of the results across model instances.

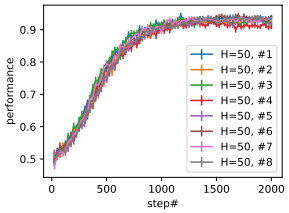
D Platform details

We used 3 computers with the following specifications: (1) 56 core CPUs (256G memory) with 4 V100 GPUs (16G memory each), (2) 56 core CPUs (256G memory) with 4 V100 GPUs (32G memory each), and (3) 96 core CPUs (256G memory) with 4 A100 GPUs (40G memory each). All code is implemented with Python (3.9.13) / Pytorch (1.12.1).

(a) Performance ramp ($H = 500$)

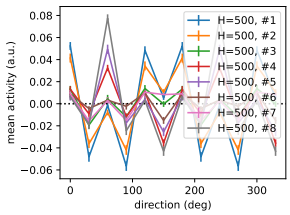


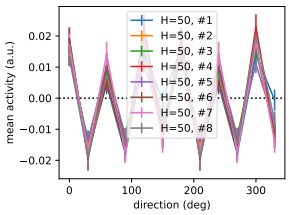
(b) Performance ramp ($H = 50$)



(c) Hexagonal modulation ($H = 500$)

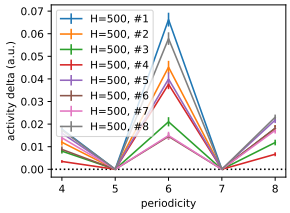
(d) Hexagonal modulation ($H = 50$)

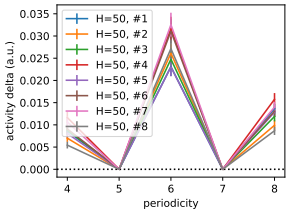




(e) Periodicity ($H = 500$)

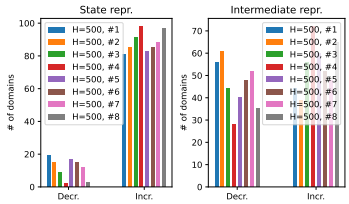
(f) Periodicity ($H = 50$)





(g) Distance coding ($H = 500$)

(h) Distance coding ($H = 50$)



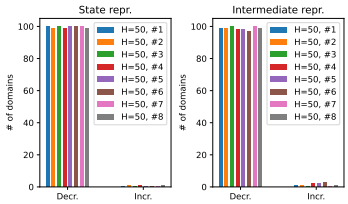


Figure 8: Results on the two-dimensional hierarchy task, analogous to Figure 5, from 8 models with larger ($H = 500$) or smaller ($H = 50$) memory.