A neural network model trained on free recall learns the method of loci

Moufan Li (moufan.li@nyu.edu)
Department of Psychology, New York University

Kristopher T. Jensen (kris.jensen@ucl.ac.uk)
Sainsbury Wellcome Centre, University College London

Marcelo G. Mattar (marcelo.mattar@nyu.edu)
Department of Psychology, New York University

Abstract

Humans preferentially recall items that are presented in close temporal proximity together – a phenomenon known as the 'temporal contiguity effect'. In this study, we investigate whether this phenomenon emerges naturally when training a recurrent neural network with episodic memory on free recall tasks. The model learns to recall items in the order they were presented, consistent with the human contiguity effect. The strength of this effect predicts the performance of individual networks, mirroring experimental findings in humans where stronger contiguity effects predict higher recall performance. The contiguity effect in the model is supported by a neural representation of item index, resembling the 'method of loci'. This differs from prominent computational models of human memory, which use a slow decay of past information to guide sequential retrieval. Our findings provide insights into the mechanisms underlying episodic memory and pave the way for future studies of its interactions with other cognitive processes.

Keywords: neural network; episodic memory; free recall; temporal contiguity effect; method of loci

Introduction

The free recall task paradigm has been widely used to study the behavioral structure of recalling a sequence of items from episodic memory. In this task, participants are first presented with a list of items and subsequently required to recall them all in any order. Prior research has found various patterns in human free recall, including serial position effects (Murdock Jr, 1962) and contiguity effects. This paper focuses on the temporal contiguity effect (Kahana, 1996; Howard & Kahana, 1999), which refers to the finding that after recalling an item, the next recall tends to be an item that is at a nearby serial position to the previously recalled item. Additionally, people exhibit a higher tendency to recall items presented after the previously recalled item than items presented before the previous item, which is known as forward asymmetry.

Several computational models have been developed to explain the temporal contiguity effect. A particularly prominent class of models is the temporal context model (TCM) (Howard & Kahana, 2002; Sederberg, Howard, & Kahana, 2008) and its successors, including the context maintenance and retrieval model (CMR) (Polyn, Norman, & Kahana, 2009). These models suggest a slowly drifting temporal context that retains information about previously studied items. Temporal contiguity in these models arises because the temporal context retrieved during memory recall serves as the cue for the next recall. Items near the previously recalled item are more likely to be recalled because their context is more similar to the cue due to the slowly drifting context during the presentation of items. TCM successfully explains both the recency effect and the temporal contiguity effect observed in human behavior.

Studies have also indicated a correlation between the temporal contiguity effect and human free recall performance, with individuals exhibiting stronger temporal contiguity effects also demonstrating better recall performance (Sederberg, Miller, Howard, & Kahana, 2010). Behaviorally, the optimal policy for free recall in a TCM- or CMR-based system is to start from the beginning of the list and recall in a forward order (Q. Zhang, Griffiths, & Norman, 2023). This is similar to the behavior associated with a mnemonic technique known as 'the method of loci' or the memory palace (Yates, 2013). People who learn to use this technique encode a list of items in well-defined locations within a mentally constructed environment, facilitating recall by following an ordered route. This strategy can significantly enhance human memory ability (Maguire, Valentine, Wilding, & Kapur, 2003).

While models of the TCM family assume that the contiguity effect results from a slowly drifting item-related temporal context, in this paper we suggest a possible distinct mechanism for the contiguity effect and, as a result, for the dynamics of human episodic memory more generally. Our approach consists of training a recurrent neural network (RNN) model augmented with episodic memory on a free recall task. This model makes minimal *a-priori* assumptions about the neural mechanisms underlying free call. Our objective is to investigate the neural dynamics in the trained model. In particular, we wish to know if the contiguity effect emerges naturally in a model trained to achieve optimal performance on this task. Furthermore, we will analyse how the temporal context is represented in this artificial neural system and compare the properties of the artificial system to free recall in humans.

Our model architecture is based on neural networks utilizing the key-value memory model (Miller et al., 2016; J. Zhang, Shi, King, & Yeung, 2017; Pritzel et al., 2017; Fortunato et al., 2019), which uses a list containing pairs of keys and values as a memory system. During memory retrieval, the model compares a 'query' to all keys, and retrieves a value according to the similarity between the keys and the query. This architecture has been proposed as a computational model of the hippocampus and used to study systems

In L. K. Samuelson, S. L. Frank, M. Toneva, A. Mackey, & E. Hazeltine (Eds.), *Proceedings of the 46th Annual Conference of the Cognitive Science Society.*

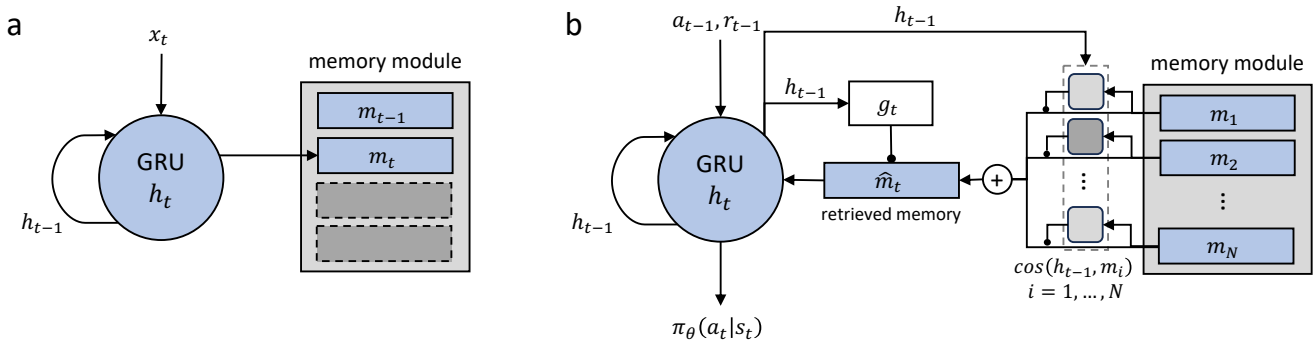

Figure 1: Model architecture. The model contains an RNN consisting of GRUs connected to a memory module. a) Encoding phase. At each time step, the model receives an item $\boldsymbol{x}_t$ as input and updates its hidden state $\boldsymbol{h}_t$. This hidden state is then appended to the memory module as a new memory $\boldsymbol{m}_t$. b) Recall phase. At each time step, the model computes the cosine similarity between the current hidden state $\boldsymbol{h}_{t-1}$ and each stored memory $\boldsymbol{m}_i$ to generate a similarity vector. A memory is retrieved by computing the weighted sum of all memories with the similarity vector as the weights. Finally, the hidden state of the GRU is updated as a function of the previous hidden state $\boldsymbol{h}_{t-1}$, the current input, and the retrieved memory $\hat{\boldsymbol{m}}_t$. A scalar memory gate g_t is also computed from the GRU hidden state to gate the retrieved memory. During the recall phase, the most recent action a_{t-1} and reward r_{t-1} are returned to the model as inputs.

and tasks that combine learning and memory (Lu, Hasson, & Norman, 2022; Whittington et al., 2020). While these studies typically use tasks where a single memory is retrieved for a given input, here we trained the model to perform sequential retrieval in a free recall task.

Our results show that the trained model exhibits the temporal contiguity effect consistent with human behavioral data, even though the training process does not explicitly encourage this behavior. However, the contiguity effect does not arise as a result of a drifting temporal context, as assumed in previous models of human episodic memory such as TCM. Instead, the trained model learns a joint representation of item index and identity during encoding, which is then reinstated during memory retrieval to recall items in forward order. Unlike in TCM, the item index representation is independent of specific item information. This representation of item index resembles the use of locations for encoding different items in the method of loci. Multiple models trained with the same hyperparameters display individual differences in recall behavior, with those learning to recall in a forward order demonstrating superior performance. Since minimal assumptions are used in the training of the model, these findings suggest that recalling memories sequentially and in the forward order – a pattern observed in free recall data from humans – might be an optimal strategy.

More broadly, our model introduces sequential memory to a neural network model that can be trained flexibly on different tasks, providing a tool to study the use of sequential memory and the underlying neural mechanisms for a wider range of tasks and cognitive functions.

Methods

Task

We used a free-recall task to train the model. In the task, the model was presented with a list of items and was then trained to recall the items in any order. Items were represented as 50-dimensional one-hot vectors, and each trial contained a list of 8 items as the input sequence. There was an encoding phase and a recall phase in each trial. During the encoding phase, the model received as inputs one item per time step. During the recall phase, the model recalled an item per time step and was required to recall all the items in the list within 8 time steps. Correct outputs resulted in a reward (+1) to the model. If the model recalled an item that was not presented during the encoding phase, or if it recalled an item more than once, the outputs were marked as incorrect and the model received a penalty (-1). The item list presented during the encoding phase was generated independently on each trial by randomly sampling 8 items from the full set of 50 items without replacement.

Model structure

The model was composed of a 'context module' and a 'memory module'. The 'context module' consisted of 128 gated recurrent units (GRU) (Cho et al., 2014). At each time step, the GRU received inputs and produced outputs with evolving dynamics according to

$$\boldsymbol{h}_t = \phi_\theta(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}) \tag{1}$$

$$\boldsymbol{y}_t = f(\boldsymbol{h}_t) \tag{2}$$

Here, ϕ denotes a forward pass through the recurrent GRU dynamics, θ denotes the set of all model parameters, $\boldsymbol{x}_t$ are inputs to the GRU, and $\boldsymbol{y}_t$ are outputs computed from the GRU

hidden state $\boldsymbol{h}_t$, which was reset at the beginning of each trial. The 'memory module' consisted of a list of slots for storing memories. At the start of each trial, the memory module was cleared to allow the storage of new memories.

Memory encoding In the encoding phase (Figure 1a), the GRU received a one-hot representation of an item in the list at each time step as input. It then updated the GRU hidden state $\boldsymbol{h}_t$ and appended it to the memory module as an episodic memory $\boldsymbol{m}_t$.

Memory retrieval In the recall phase, the model retrieved a memory from the memory module and produced an output at every time step (Figure 1b). After retrieving a memory, the GRU took the retrieved memory as part of its inputs and updated its hidden state based on the retrieved memory as well as other inputs and the previous hidden state. The output consisted of a 50-dimension policy $\pi_\theta(a_t)$, which was a set of probabilities associated with each possible item to recall.

The memory retrieval process was as follows. At the start of each time step, the model computed the cosine similarity between the current hidden state $\boldsymbol{h}_{t-1}$ and each of N memories stored in the memory module $\{\boldsymbol{m}_i | i = 1, ..., N\}$ to produce a similarity vector $\boldsymbol{s}$,

$$s_i = cos(\boldsymbol{h}_{t-1}, \boldsymbol{m}_i), \qquad i = 1, ..., N. \tag{3}$$

The model then computed the retrieved memory as the weighted sum of all the stored memories according to the weights of the similarity vector. To favor the retrieval of a single memory rather than a combination of multiple memories, we first passed the similarity vector through a softmax function with a low temperature ($\tau = 0.1$),

$$\hat{s_i} = \frac{e^{\frac{s_i}{\tau}}}{\Sigma_{i=1}^{N} e^{\frac{s_i}{\tau}}}. \tag{4}$$

Then we computed the retrieved memory $\hat{\boldsymbol{m}}_t$ by treating $\hat{\boldsymbol{s}} = [\hat{s_1}, \hat{s_2}, ..., \hat{s_N}]$ as weights for each memory,

$$\hat{\boldsymbol{m}}_t = \Sigma_{i=1}^{N} \hat{s_i} \boldsymbol{m}_i. \tag{5}$$

This weighted averaged memory was used instead of the single most similar memory to make the memory retrieval process differentiable during training.

We also included a scalar memory gate g_t in the model to gate the retrieved memory before adding it to the hidden state of the GRU. This allows the model to flexibly choose whether to use the episodic memory to help it perform the task. g_t was generated from the current hidden state of the GRU $\boldsymbol{h}_{t-1}$,

$$g_t = \sigma(U_g \boldsymbol{h}_{t-1} + b_g). \tag{6}$$

Here, U_g and b_g are the weights and bias of a linear layer. We observed that g_t had values close to 1 in the free recall task, suggesting that the model used a newly retrieved memory at each time step to solve the task, instead of retrieving a single memory containing the information of all the items at the beginning of the recall phase. Finally, the retrieved memory as an input the the GRU was $g_t \cdot \hat{\boldsymbol{m}}_t$.

To force the model to use previously stored episodic memories in the memory module instead of working memory stored in the hidden state, the hidden state of the GRU was reset to a random vector between the encoding and recall phases. We also allowed the model to update its hidden state for a single time step without any recall before retrieving the first memory, so that it could learn an initial state for retrieving the first memory.

Training

We trained the model with the advantage actor-critic (A2C) reinforcement learning algorithm (Mnih et al., 2016; Jensen, 2023). The last reward r_{t-1} and last action a_{t-1} sampled from the policy $\pi_\theta(a_{t-1})$ were returned to the model as inputs at the next time step (Wang et al., 2016). This was important for the model to perform the task, since it provided information about what items have been previously recalled, and therefore also which items have yet to be recalled. In the default setting, the temporal discount factor of reward was set to 0.99.

To encourage the model to recall only one memory at a time and inhibit the retrieval of other memories, we added another entropy regularization term $H_m = -\sum_{i=1}^{N} \hat{s_t}(i) \log \hat{s_t}(i)$ on the recall weights $\hat{\boldsymbol{s}}$ in the loss function. This term encouraged the model to decrease the entropy of the memory similarity $\hat{\boldsymbol{s}}$, which made it closer to a one-hot vector. As a consequence, the retrieved memory $\hat{\boldsymbol{m}}_t$ was closer to a single memory in the memory module instead of a combination of multiple memories. A hyper-parameter β controls the weight of the regularization term in the loss.

Results

Temporal contiguity effect in model behavior

The model exhibited a strong temporal contiguity effect after training (Figure 2a). It showed a higher tendency to recall an item that is closer to the last recalled item in the list, despite the training loss not enforcing this particular strategy. It also had a higher probability of recalling items in the forward order of presentation than in reverse order. Compared to human behavior (Kahana, 1996), the model showed a higher forward asymmetry. It exhibited a very high tendency to recall the item that was presented right after the just recalled item in the list and often recalled the whole list in forward order.

Neural mechanism of the contiguity effect

We next investigated the neural mechanism driving the contiguity effect in our model. When quantifying the cosine similarity between the hidden states of the model at the encoding phase and the recall phase, we found that states at the same time step within each phase had high similarities. Additionally, the similarity smoothly decayed as a function of the distance between two items in the memory list (Figure 2b), reminiscent of the temporal context model.

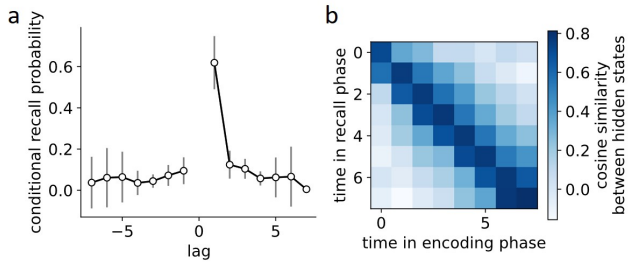

Figure 2: Model behavior and similarity between hidden states. a) Conditional recall probability of each item as a function of the relative position from the most recently recalled item in the presented list (lag). The figure shows the average of 20 models with the same hyperparameters but trained with different random seeds. The error bars indicate the standard deviation across random seeds. b) Cosine similarity between each GRU hidden state in the encoding phase and each hidden state in the recall phase. For the encoding phase, we used the hidden state after receiving an item as input at each time step. For the recall phase, we used each hidden state before retrieving a memory (i.e., the hidden state that determines the memory to be retrieved).

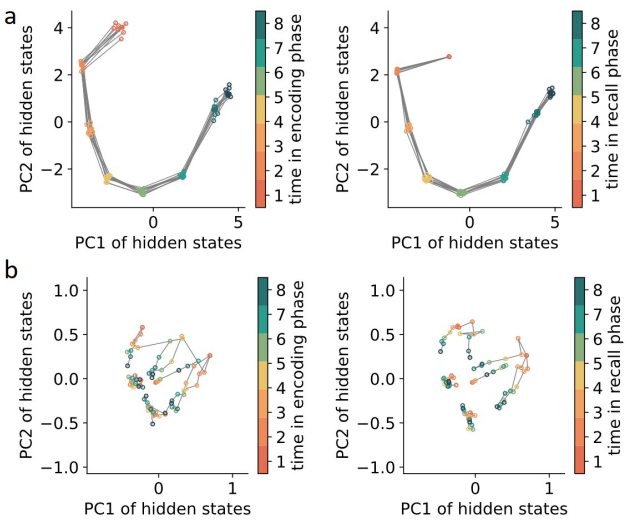

Figure 3: PCA plots of the neural network memory model and TCM. a) The first two PCs of the GRU hidden states of the neural network memory model during the encoding phase (left) and the recall phase (right). b) The first two PCs of the context of TCM during the encoding phase (left) and the recall phase (right). Dots with different colors represent hidden states at different time steps, and each trajectory represents a different trial. A single PCA model was fitted to activity from both the encoding and recall phases for each model. Each plot includes trajectories from 10 trials.

However, we also found notable differences between the neural representation of this model and TCM. To illustrate this, we plotted the first two principal components (PC) of the hidden state in Figure 3a. The hidden state of the RNN followed very similar trajectories across trials in both the encoding and recall phases, regardless of what specific items were presented in the input sequence. Instead of a slowly drifting context that carries information about nearby items, our model thus learned dynamics that faithfully encode the time within the task, with nearby time points exhibiting more similar representations. This allowed the model to sequentially reinstate each hidden state from the encoding phase to retrieve the correct memories.

We made similar plots for TCM in Figure 3b as a comparison. As the temporal context in TCM depended on specific item information that varies across trials, there was a significant distinction in the trajectories of different trials. This suggests an important difference between TCM and our proposed model, with TCM performing associative recall, while our model using an index representation to recall each item from the presentation phase. Compared to TCM, the mechanism of our model is more similar to the method of loci, with the representation of index in the model corresponding to the locations where individual items are stored in the method of loci.

While these prototypical trajectories faithfully encode item position, this does not imply that no information is preserved about item identity. When training a decoder to predict item identity from the hidden states (Figure 4a, b), we found that a hidden state contained the most information about the item that was encoded or recalled at the current time step, and that the information gradually decreased as other memories were encoded or retrieved in the following time steps. In the encoding phase, information about the first presented item persisted much longer in the model's hidden state than information about the other items. Meanwhile, information about the item *index* (as opposed to item identity) was always maintained in hidden states during both the encoding and recall phases (Figure 4c). These findings indicate that the model stores both information about the item identity and the item index.

To further study how these two kinds of information were encoded in the hidden states, we performed decoding of item identity and item index from an increasing number of PCs of the hidden states (Figure 4d). The decoding accuracy of the item index rose quickly as the first few PCs were included for decoding, while the decoding accuracy of the item identity increased more slowly with the number of PCs. These results suggest that item index was encoded in a low-dimensional hidden state space, and that item identity was encoded in a higher-dimensional space. They also indicate that the model relied primarily on the index code to guide memory recall, since the first few PCs explained a large fraction of variance in the hidden states. However, the index code coexisted with the item identity code that occupied subsequent PCs. This suggests that there could also be item-related temporal context in our model, though unlike TCM, item information did

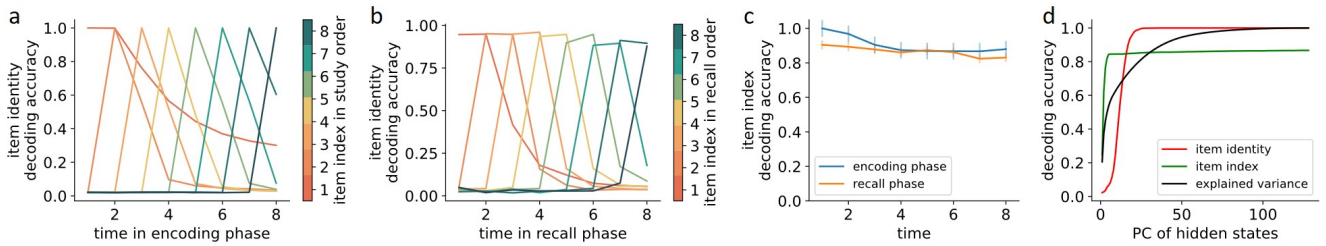

Figure 4: a) Decoding accuracy of item identity from hidden states during the encoding phase. Each curve represents items presented at a particular time step. Each data point in the plot is generated with a different decoder, which is trained with cross-validation on data from 1000 trials. b) Decoding accuracy of item identity from hidden states during the recall phase. c) Decoding accuracy of item index from hidden states during either the encoding or recall phase. Data from all time steps was combined to train the decoder. d) Decoding accuracy of item identity (red) and item index (green) from an increasing number of PCs of the hidden states, plotted together with the cumulative explained variance of the PCs (black). All results in this figure are averaged over 20 models with different random seeds. Error bars in panel c and d indicate standard deviation across random seeds.

not decay exponentially with time in our model (Figure 4a, b), and the model did not rely on item identity to perform memory recall.

Individual differences between models

Models with the same hyperparameter setting but different random seeds showed different levels of contiguity effect (Figure 5a). The model with the highest forward asymmetry (seed 1 in Figure 5a) almost always recalled items in forward order. Other models (seed 2 and 3) exhibited more smoothly changing conditional recall probabilities but still some degree of temporal contiguity, while some (seed 4) showed little to no temporal structure in their recall patterns. We hypothesized that this variability across models might relate to the variability between individual humans in similar free recall tasks.

To further study factors related to individual differences between models, we used two metrics to quantify the contiguity effect of a model, the 'forward asymmetry' and the 'temporal factor'. The forward asymmetry (FA) evaluates the tendency of a model to recall an item presented after the previously recalled item instead of recalling an item presented before the previous item. FA is defined as the proportion of forward transitions in all recall transitions, where a recall transition is defined as a pair of consecutively recalled instances. The temporal factor (TF) is introduced by Sederberg et al. (2010) and quantifies the tendency of an agent to recall an item that is in close temporal proximity to the previously recalled item. TF computes a score for each recall transition. The score is higher when two consecutively recalled items are closer to each other in the presentation list, without considering the forward or backward order of the recall transition. These two metrics together quantify the level of temporal contiguity exhibited by a model.

We found that models with higher forward asymmetry (Spearman correlation, $r^2 = 0.89$) and temporal factor (Spearman correlation, $r^2 = 0.96$) exhibited better task performance (Figure 5c, d). Similar patterns have also been observed in human behavior (Q. Zhang et al., 2023; Sederberg et al., 2010). While the task performance of different models varied from around 70% to near 100%, models that learned to always recall in a forward order almost all exhibited near-perfect performance. This suggests a normative explanation for the observed pattern of forward recall in both humans and artificial neural networks.

We proceeded to investigate what factors influenced the model to learn a strategy of forward recall. Human behavioral experiments have shown that when people are asked to try their best to recall the whole list, they are more likely to start from the beginning of the list and perform forward recall. When people are asked to only recall a few items, they will tend to recall from the end of the list, which they have the most context about (Tan, Ward, Paulauskaite, & Markou, 2016). A similar pattern was observed in our model. By varying the temporal discount factor of the reward, γ, we could modify the degree to which future reward was prioritized compared to immediate reward when training the model. In particular, a high value of γ (i.e. little temporal discounting) during training can encourage the model to optimize for recalling the entire list of items rather than greedily recalling a single correct item at any given time step. Consistent with prior human experiments, we found that a higher γ led to both higher forward asymmetry and higher temporal factor (Figure 5d).

Discussion

In this study, we designed a neural network model capable of performing sequential memory retrieval by using the dynamically changing hidden state of the recurrent unit as a memory retrieval cue and integrating the retrieved memory into the recurrent dynamics. After training the model on a free recall task, it exhibited a temporal contiguity effect reminis-

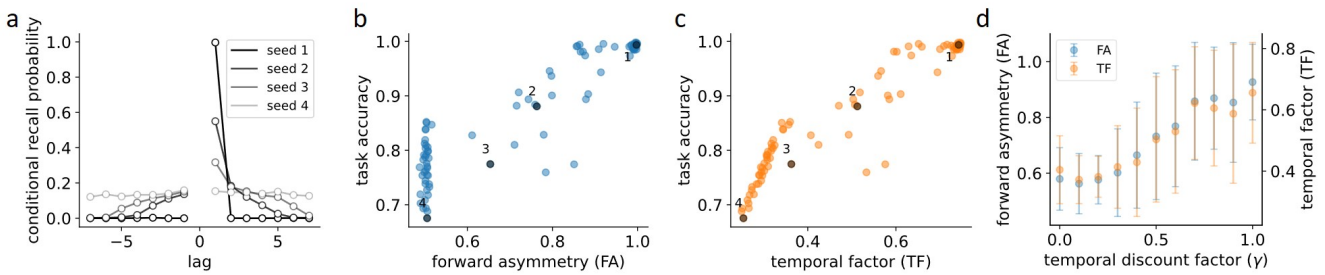

Figure 5: Individual differences in contiguity effects and related factors. a) Different levels of contiguity effect exhibited by models with the same hyperparameters and different random seeds. All models in this plot are trained with the same temporal discount factor $\gamma = 0.6$. b) Task accuracy of models with different forward asymmetry. Each dot represents a model with a different random seed. c) Task accuracy of models with different temporal factors. Panels b and c show data from 100 models trained with the same temporal discount factor of $\gamma = 0.6$. The four darker data points (1-4) correspond to the 4 models shown in panel a. d) Forward asymmetry (FA) and temporal factor (TF) of models trained with different temporal discount factors ($\gamma = 0.0, 0.1, 0.2, ..., 1.0$). Dots and error bars indicate mean and standard deviation across 20 random seeds. In general, larger γ led to more forward recall (higher FA) and increased sequential recall of items in close temporal proximity in the presented list (higher TF).

cent of that observed in human experimental data. The model learned a representation of the item index to help it recall the items in a forward order. This mechanism is different from the slowly drifting item-related temporal context in TCM. Instead, it aligns more with the method of loci technique, which visits an ordered route to recall items that are associated with locations in a pre-constructed internal environment. We also observed a positive correlation between task performance and the strength of the contiguity effect across models, with models favoring forward recall exhibiting superior performance.

Our findings suggest that recalling in a forward order is a normative solution for free recall in recurrent neural networks. While previous studies have discussed the optimality of forward recall (Q. Zhang et al., 2023), these studies have focused on the TCM class of models, which assume the existence of a slowly changing representation of item identity used to drive recall. In contrast, our model autonomously learned the temporal contiguity effect without any prior assumptions on possible mechanisms, indicating that the strategy of forward recall could be advantageous for free recall more generally. The emergence of an index code in our model also suggests an advantage of learning some item-independent scaffold for encoding items during free recall, consistent with the structure in the method of loci.

While our recurrent memory model learned a mechanism reminiscient of the method of loci, our findings are not mutually exclusive with the temporal context model. Decoding results still suggest a drifting temporal context related to item identity information, though models that learned the method of loci mainly rely on the index code instead of this item-related temporal context to perform the task. This could be a result of training the model specifically on the free recall task, which may not necessarily need an item-related temporal context. However, item-related temporal context could be important in other settings, such as when there is interference of memories across episodes, and when there are semantic connections between consecutive events. To investigate the utility of item-related temporal context in more flexible RNNs, it would be interesting to train our model on tasks where such contextual information is more important. As we have observed individual differences between models in this study, further analysing models that did not learn the method of loci could also shed more light on the diversity of strategies used for memory encoding and retrieval in humans and artificial systems.

Importantly, our model replicates the patterns of sequential memory retrieval observed in humans while maintaining the flexibility of a neural network to learn tasks of varying complexities. It therefore provides a tool for studying the use of sequential episodic memory in other cognitive tasks, such as decision-making, planning, and inference. As an example, we could train the neural network memory model on tasks requiring both spatial and temporal information and investigate the consequences of these task features on the learned representations. Additionally, decision-making and planning tasks usually require subjects to consider multiple future states. If episodic memory is needed for a planning process, it could therefore be natural to also retrieve a sequence of memories for simulating the future outcomes resulting from multiple actions. Extending our model to more demanding tasks involving such higher-order cognitive processes is an interesting and important avenue for future work.

References

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using rnn encoder-

decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Fortunato, M., Tan, M., Faulkner, R., Hansen, S., Puigdomènech Badia, A., Buttimore, G., ... Blundell, C. (2019). Generalization of reinforcement learners with working and episodic memory. *Advances in neural information processing systems*, *32*.

Howard, M. W., & Kahana, M. J. (1999). Contextual variability and serial position effects in free recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *25*(4), 923.

Howard, M. W., & Kahana, M. J. (2002). A distributed representation of temporal context. *Journal of mathematical psychology*, *46*(3), 269–299.

Jensen, K. T. (2023). An introduction to reinforcement learning for neuroscience. *arXiv preprint arXiv:2311.07315*.

Kahana, M. J. (1996). Associative retrieval processes in free recall. *Memory & cognition*, *24*(1), 103–109.

Lu, Q., Hasson, U., & Norman, K. A. (2022). A neural network model of when to retrieve and encode episodic memories. *elife*, *11*, e74445.

Maguire, E. A., Valentine, E. R., Wilding, J. M., & Kapur, N. (2003). Routes to remembering: the brains behind superior memory. *Nature neuroscience*, *6*(1), 90–95.

Miller, A., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., ... Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International conference on machine learning* (pp. 1928–1937).

Murdock Jr, B. B. (1962). The serial position effect of free recall. *Journal of experimental psychology*, *64*(5), 482.

Polyn, S. M., Norman, K. A., & Kahana, M. J. (2009). A context maintenance and retrieval model of organizational processes in free recall. *Psychological review*, *116*(1), 129.

Pritzel, A., Uria, B., Srinivasan, S., Badia, A. P., Vinyals, O., Hassabis, D., ... Blundell, C. (2017). Neural episodic control. In *International conference on machine learning* (pp. 2827–2836).

Sederberg, P. B., Howard, M. W., & Kahana, M. J. (2008). A context-based theory of recency and contiguity in free recall. *Psychological review*, *115*(4), 893.

Sederberg, P. B., Miller, J. F., Howard, M. W., & Kahana, M. J. (2010). The temporal contiguity effect predicts episodic memory performance. *Memory & cognition*, *38*, 689–699.

Tan, L., Ward, G., Paulauskaite, L., & Markou, M. (2016). Beginning at the beginning: Recall order and the number of words to be recalled. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *42*(8), 1282.

Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., ... Botvinick, M. (2016). Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*.

Whittington, J. C., Muller, T. H., Mark, S., Chen, G., Barry, C., Burgess, N., & Behrens, T. E. (2020). The tolman-eichenbaum machine: unifying space and relational memory through generalization in the hippocampal formation. *Cell*, *183*(5), 1249–1263.

Yates, F. A. (2013). *Art of memory*. Routledge.

Zhang, J., Shi, X., King, I., & Yeung, D.-Y. (2017). Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on world wide web* (pp. 765–774).

Zhang, Q., Griffiths, T. L., & Norman, K. A. (2023). Optimal policies for free recall. *Psychological Review*, *130*(4), 1104.