Brain-Inspired Architectures for Efficient and Meaningful Learning from Temporally Smooth Data

Anonymous Author(s)
Affiliation
Address
email

Abstract

How can learning systems exploit the temporal smoothness of real-world training data? We tested the learning of neural networks equipped with two architectural features inspired by the temporal properties of neural circuits. First, because brain dynamics are correlated over time, we implemented a leaky memory mechanism in the hidden representations of neural networks. Second, because cortical circuits can rapidly shift their internal state, "resetting" their local memory, we implemented a gating mechanism that could reset the leaky memory. How do these architectural features affect learning efficiency and how do they affect the representations that are learned by neural networks? We found that networks equipped with leaky memory and gating could exploit the temporal smoothness in training data, surpassing the performance of conventional feedforward networks. Moreover, networks with multi-scale leaky memory and gating could learn internal representations that "unmixed" data sources which vary on fast and slow timescales across training samples. Altogether, we showed that brain-inspired architectural mechanisms enabled neural networks to learn more efficiently from temporally smooth data, and to generate internal representations that separate timescales in the training signal.

1 Introduction

Events in the world are correlated in time: the information that we receive at one moment is usually similar to the information that we receive at the next. For example, when having a conversation with someone, we see multiple samples of the same face from different angles over the course of several seconds (Figure 1.A). What characteristics may enable a learning system to exploit this temporal smoothness?

Neural circuits have temporal characteristics which may allow them to take advantage of the temporal properties of information for more efficient and meaningful representation learning. Cortical dynamics exhibit *autocorrelation* on the scale of milliseconds to many seconds (Murray et al., 2014; Honey et al., 2012; Bright et al., 2020). Such autocorrelation is observed even in the absence of external stimuli, suggesting that correlation in consecutive internal states is unavoidable (Murray et al., 2014; Raut et al., 2020). One possible benefit of such autocorrelation is that it may enable learning systems to combine information from consecutive training samples. But cortical states are not always correlated: neural circuits can identify *event boundaries* in the information and shift their state accordingly. This shift appears to be associated with "resetting" of context representations (DuBrow et al., 2017; Chien and Honey, 2020; Baldassano et al., 2018). This "memory resetting" mechanism may enable neural circuits to flexibly adapt its learning, only combining information over time for related training samples.

We hypothesized that a combination of these two brain-inspired mechanisms – leaky memory and memory gating – could enable neural networks to flexibly learn from different amounts of temporal

Submitted to 2nd Workshop on Shared Visual Representations in Human and Machine Intelligence (SVRHM).

smoothness in the training data. First, we tested whether these two brain-inspired mechanisms would enable a neural network to learn more efficiently from temporally smooth training data. Second, we studied the nature of the internal representations learned by networks equipped with these brain-inspired mechanisms.

2 Brain-inspired mechanisms for learning from smooth data

Leaky memory: We added leaky memory to the internal representations (hidden units) by linearly mixing them across consecutive time points. Hidden unit activations were updated according to following function:

$$H(n) = \alpha H(n-1) + (1-\alpha)ReLU(W_{IH}I(n)) \tag{1}$$

where $H(n)$ is the state of the hidden units for trial n, $I(n)$ is the state of the input units for trial n, α is a leak parameter, W_{IH} are the connections from the input layer to the hidden layer, and ReLU is a rectified linear activation. We set α = 0.5 for modeling leaky memory in these experiments.

Memory Gating: In order to reduce the interference between unrelated information in the leaky memory, we employed a gating mechanism to reset the memory. Memory was reset (setting α = 0 in Eq(1)) at the transitions between categories in classification tasks (Figure 1.E) and at the transitions between dissimilar features in reconstruction tasks (Figure 2.B).

3 Learning efficiency in brain-inspired architectures

We first explored how these brain-inspired mechanisms affected the speed and accuracy of category learning, for training data with varying levels of smoothness.

3.1 Methods

We tested MNIST, Fashion-MNIST, and further synthetic datasets containing low category overlap (LeCun et al., 2010; Xiao et al., 2017). We trained models using backpropagation with mean squared error (MSE) primarily for the ease of comparison with later reconstruction error measures in this manuscript. To test incremental learning, we employed stochastic gradient descent (SGD), updating weights for each training sample. We applied ReLU to hidden units and Softmax or Sigmoid to the output units. For initialization and optimization methods see Appendix A.1.
Hyperparameters. For MNIST and Fashion-MNIST, we used a 3-layer fully connected network with (784, 392, 10) dimensions and a learning rate of 0.01. We used the same learning rate across all conditions so that the smoothness would be the only variable manipulated across conditions. For hyperparameters in synthetic dataset see Appendix A.1.
Manipulating smoothness in training data. We manipulated smoothness in the training data by varying the number of consecutive samples drawn from the same category. To sample with minimum smoothness, we sampled exactly one exemplar from each category (Figure 1.B). This condition is called "minimum smoothness" because all consecutive items were from different categories, and there were not more examples from a category until all other categories were sampled. We increased smoothness by increasing the number of consecutive samples drawn from each category (e.g. 3 repetitions and 5 repetitions in Figure 1.B). Finally, we also used the standard random sampling method, in which items were sampled at random, without replacement, from the training set. The training set was identical across all conditions, as was the order in which samples were drawn from within a category (Figure 1.B).

3.2 Results

Learners with leaky memory learned more efficiency from temporally smooth data (Figure 1.D, E). Conversely, in memoryless learners, smoothness slowed learning (Figure 1.C). Moreover, adding a gating mechanism to the leaky memory units further increased their learning efficiency. In learners with leaky memory and gating, all levels of smoothness significantly outperformed memoryless learners (Figure 1.E). These findings generalized across MNIST, Fashion-MNIST, and synthesized datasets (see Appendix A.1).

3.3 Discussion

In contrast to memoryless learners, learners with leaky memory could exploit the shared information across samples for more efficient category learning. Importantly, the resetting mechanism prevented the mixing of hidden representations from samples of different categories; this allowed the leaky memory systems to benefit most from the smoothness, while not suffering from between-category interference. Across all levels of smoothness in training data, networks with leaky memory and resetting surpassed the performance of feedforward networks, resulting in more efficient learning (Figure 1.E). This is notable because the leaky memory is easy to implement, and autocorrelated states are ubiquitous in brain dynamics (Honey et al., 2012; Murray et al., 2014).

4 Representations learned by brain-inspired architectures

In the real world, we may need to learn from data with multiple levels of smoothness. For instance, while having a conversation, features around a person's mouth may change quickly, while their face outline changes more slowly (Figure 2.A). Moreover, there are no labels to support the learning of representations in this setting. We hypothesized that neural networks equipped with multi-scale (i.e. fast and slow) leaky memory and gating could learn to effectively represent structures that vary on multiple scales.

4.1 Methods

Dataset. We synthesized a simplified training dataset which contained three levels of temporal structure. The input to the model at each time point contained 3 subcomponents (top row, middle row, bottom row),which varied at fast, medium and slow timescales, respectively (see Appendix A.2)).
Architectures. We used the same brain-inspired mechanisms for unsupervised learning models: leaky memory and gating. To evaluate the effectiveness of the added mechanisms, we compared 4 types of autoencoders (AE): i) Feedforward AE (Figure 2.C); ii) AE with leaky memory in internal representations (Figure 2.D); iii) AE with multi-scale leaky memory in internal representations (Figure 2.E), inspired by the presence of multiple time-scales within a single neural circuit (Bernacchia et al., 2011; Ulanovsky et al., 2004); iv) AE with multi-scale leaky memory in internal representations and feature-boundary gating (Figure 2.F).
Hyperparameters. To vary the timescale of leaky memory, we varied the time constants across the nodes in the hidden layer. Thus, the variable α in Eq.(1) was set to 0, 0.3, and 0.6 for "no-memory", "short-memory", and "long-memory" nodes, respectively (Figure 2.G). Also, see Appendix A.2.
Un-mixing Measures. We measured the system's ability to "un-mix" the time-scale of input, i.e. to learn representations that selectively track distinct sources used to generate each training sample. In other words, we tested whether no-memory, short-memory and long-memory nodes would track the fast-, medium-, and slow-changing data features, respectively. To this end we measured the Pearson correlation between each hidden unit (no-memory, short-memory, or long-memory) and all of the data features (fast, medium and slowly changing). We then quantified the "timescale-matching" – e.g. whether the long-memory node was correlated with the slowly-varying data feature (Figure 2.H) – and the "timescale-selectivity" – e.g. whether the long-memory node was more correlated with slowly-varying features than the other features (Figure 2.I).

4.2 Results

Multi-scale networks with leaky memory and gating most effectively un-mixed fast and slow data sources: their individual hidden state units were most strongly correlated with their corresponding data features (Figure 2.H, e.g. long-memory nodes correlated most strongly with slowly-varying data), and most selective (Figure 2.I, e.g. the long-memory node was more correlated with the slow features than with the other features).

4.3 Discussion

The autoencoder model with multi-scale leaky memory and feature-boundary gating was most successful in learning internal representations which tracked distinct timescales of the input. Slowly (or quickly) varying features were extracted by slowly (or quickly) varying subsets of the network,

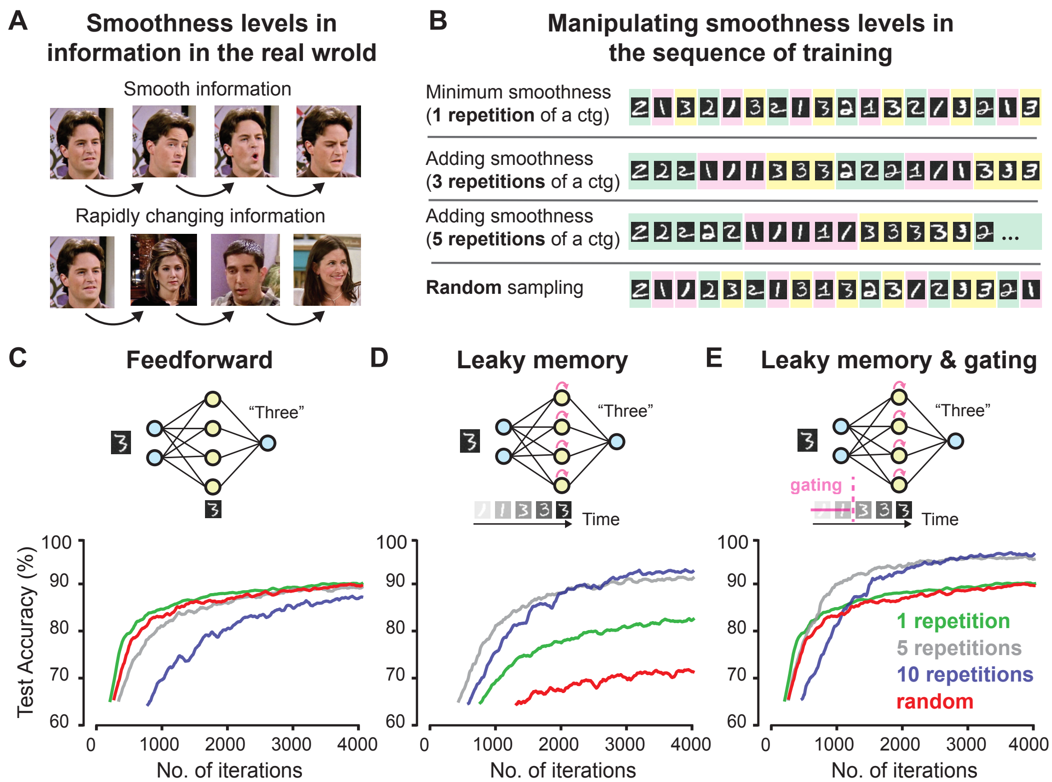

Figure 1: **Efficiency of category learning from temporally smooth data.** A) Various levels of smoothness in real-world information. B) Manipulating smoothness levels in training data. C), D), and E) show test accuracy in feedforward network, network with leaky memory in internal representations, and network with leaky memory and gating, respectively. (Curves are averaged of 5 runs with different initialization and are further smoothed using 100-iteration moving average).

analogous to a matched filter (see also Mozer (1992)). Thus, by adding leaky memory and memory-gating to a simple feedforward AE model, we equip it with an ability to separate different levels of structure in the environment.

5 Conclusion

We investigated how brain-inspired mechanisms – leaky memory and memory-gating – affected the efficiency of learning and the type of representations that were learned. We focused on settings in which the training data exhibited varying levels of temporal smoothness. We found that learners with leaky memory in internal representations and gating mechanisms were able to flexibly adapt to the smoothness in the data, so that they could benefit from repeating structure while not mixing unrelated information. Moreover, neural networks with multi-scale memory and feature-sensitive gating learned representations that un-mixed features varying on different timescales.

Features that change on different timescales may correspond to different levels of structure in the world (Wiskott and Sejnowski, 2002). Thus, the "un-mixed" representations learned by brain-inspired architectures may provide a more "meaningful" description of the input data, reflecting underlying data sources that operate on fast and slow timescales (Mitchell, 2020; Mahto et al., 2020). Moreover, because intrinsic brain dynamics vary on multiple scales (Stephens et al., 2013; Murray et al., 2014; Honey et al., 2012; Raut et al., 2020), slowly-varying brain circuits may be biased to extract slowly-varying structure from the world (Honey et al., 2017).

Leaky memory networks produced more efficient learning and more interpretable representations, even though the networks were trained with a learning rule that did not employ any temporal

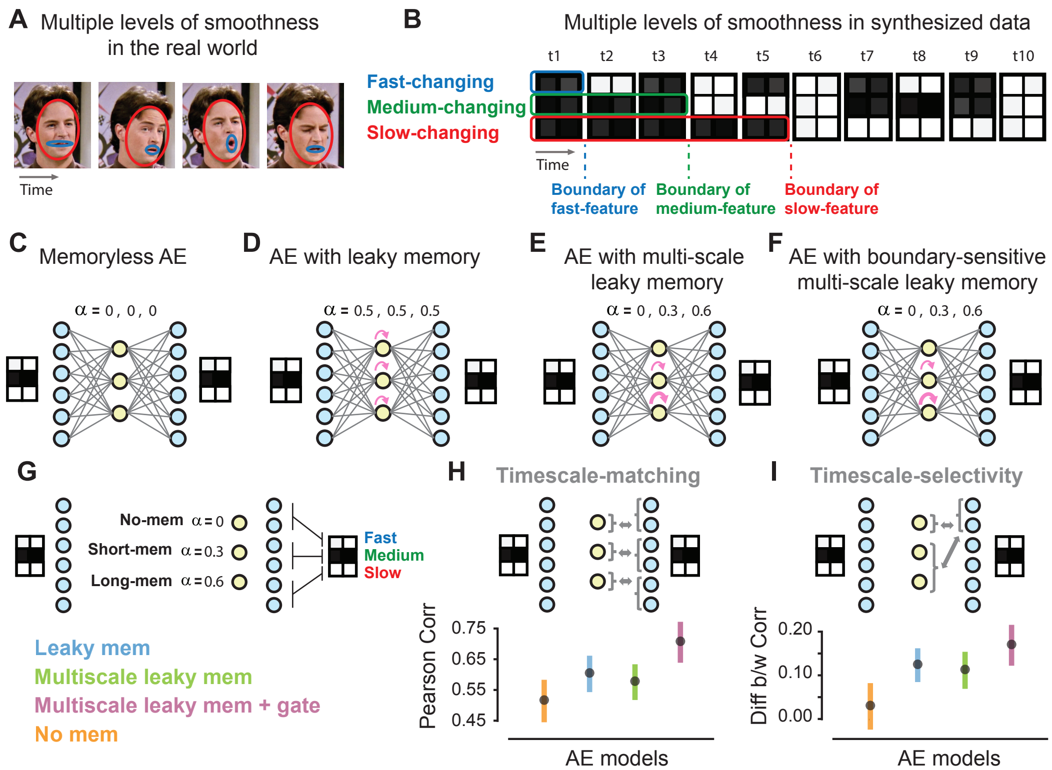

Figure 2: **Representations learned from temporally smooth data.** A) Multiple levels of smoothness in the world. B) Multiple levels of smoothness in synthesized data. C, D, E, F) Different tested AE models. G) Architectural details used for timescale-matching and timescale-selectivity analyses in part H and I. H) "Timescale-matching" of models, as measured by the squared Pearson correlation of internal representations (hidden units) with corresponding output units. I) "Timescale-selectivity" of models, measured by computing the difference between the squared Pearson correlations for time-scale matching units and non-matching units, e.g. long-memory correlation with slow features minus long-memory correlation with fast and medium features. In no-memory and in uniform leaky-memory systems, we measured these correlations for hidden units in the corresponding position as those in the multi-scale leaky memory. We ran 20 runs with different initializations. Error bars show the mean and standard deviation across 10,000 bootstraps, with 20 values per bootstrap.

information. Architectures with leaky memory and gating can thus exploit temporal structure in a way that is computationally simpler and more biologically plausible than backpropagation through time (Sutskever, 2013; Lillicrap and Santoro, 2019). The leaky-memory-plus-gating system worked well even for autoencoders, for which there are simple activation-based learning rules that do not require the propagation of partial derivatives (Lee et al., 2015).

Future work should test how leaky memory affects the learned representational space. For example, human internal representations of natural sensory input sequences appear to be smooth in time, in contrast to the representations of most feedforward nets (Hénaff et al., 2019); training neural networks with smooth data and leaky memory could potentially capture this effect. In ongoing work, we are testing whether these results generalize to larger architectures and datasets; we expect the results to have some generality, because we used simple architectures and made few domain-specific assumptions.

In sum, we tested brain-inspired architectures in learning representations from data with various amounts of temporal autocorrelation and found that such architectures enabled networks to learn more quickly from smooth data and to generate internal representations that separate distinct timescales of the data.

Broader Impact

This section is not applicable to the current work.

References

Christopher Baldassano, Uri Hasson, and Kenneth A Norman. Representation of real-world event schemas during narrative perception. *Journal of Neuroscience*, 38(45):9689–9699, 2018.

Alberto Bernacchia, Hyojung Seo, Daeyeol Lee, and Xiao-Jing Wang. A reservoir of time constants for memory traces in cortical neurons. *Nature Neuroscience*, 14(3):366–372, March 2011. ISSN 1097-6256, 1546-1726. doi: 10.1038/nn.2752. URL `http://www.nature.com/articles/nn.2752`.

Ian M Bright, Miriam LR Meister, Nathanael A Cruzado, Zoran Tiganj, Elizabeth A Buffalo, and Marc W Howard. A temporal record of the past with a spectrum of time constants in the monkey entorhinal cortex. *Proceedings of the National Academy of Sciences*, 117(33):20274–20283, 2020.

Hsiang-Yun Sherry Chien and Christopher J Honey. Constructing and forgetting temporal context in the human cerebral cortex. *Neuron*, 2020.

Sarah DuBrow, Nina Rouhani, Yael Niv, and Kenneth A Norman. Does mental context drift or shift? *Current opinion in behavioral sciences*, 17:141–146, 2017.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

Charles R Harris, K Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.

Olivier J Hénaff, Robbe LT Goris, and Eero P Simoncelli. Perceptual straightening of natural videos. *Nature neuroscience*, 22(6):984–991, 2019.

Christopher J Honey, Thomas Thesen, Tobias H Donner, Lauren J Silbert, Chad E Carlson, Orrin Devinsky, Werner K Doyle, Nava Rubin, David J Heeger, and Uri Hasson. Slow cortical dynamics and the accumulation of information over long timescales. *Neuron*, 76(2):423–434, 2012.

Christopher J Honey, Ehren L Newman, and Anna C Schapiro. Switching between internal and external modes: a multiscale learning principle. *Network Neuroscience*, 1(4):339–356, 2017.

Bernd Illing, Wulfram Gerstner, and Johanni Brea. Biologically plausible deep learning—but how far can we go with shallow networks? *Neural Networks*, 118:90–101, 2019.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 498–515. Springer, 2015.

Timothy P Lillicrap and Adam Santoro. Backpropagation through time and the brain. *Current opinion in neurobiology*, 55:82–89, 2019.

Shivangi Mahto, Vy A. Vo, Javier S. Turek, and Alexander G. Huth. Multi-timescale representation learning in lstm language models, 2020.

Melanie Mitchell. On crashing the barrier of meaning in artificial intelligence. *AI Magazine*, 41(2): 86–92, 2020.

Michael C Mozer. Induction of multiscale temporal structure. In *Advances in neural information processing systems*, pages 275–282, 1992.

John D Murray, Alberto Bernacchia, David J Freedman, Ranulfo Romo, Jonathan D Wallis, Xinying Cai, Camillo Padoa-Schioppa, Tatiana Pasternak, Hyojung Seo, Daeyeol Lee, et al. A hierarchy of intrinsic timescales across primate cortex. *Nature neuroscience*, 17(12):1661–1663, 2014.

Ryan V Raut, Abraham Z Snyder, and Marcus E Raichle. Hierarchical dynamics as a macroscopic organizing principle of the human brain. *Proceedings of the National Academy of Sciences*, 117 (34):20890–20897, 2020.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Greg J Stephens, Christopher J Honey, and Uri Hasson. A place for time: the spatiotemporal structure of neural dynamics during natural audition. *Journal of neurophysiology*, 110(9):2019–2026, 2013.

Ilya Sutskever. *Training recurrent neural networks*. University of Toronto Toronto, Canada, 2013.

T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

Nachum Ulanovsky, Liora Las, Dina Farkas, and Israel Nelken. Multiple time scales of adaptation in auditory cortex neurons. *Journal of Neuroscience*, 24(46):10440–10453, 2004.

Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, 14(4):715–770, 2002.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A Appendix

A.1 Further details about testing the learning efficiency in brain-inspired architectures

Loss functions
For the results reported in Figure 1, we used an MSE loss function, mainly for the ease of comparison with reconstruction error measures in this manuscript. Additionally, it has been shown MSE loss provides comparable performance to commonly utilized classification models with cross-entropy (CE) loss function (Illing et al., 2019).

However, we also tested memoryless classifier models with cross-entropy loss and found the same pattern: smoothness in training data slowed learning, and the condition with minimum smoothness showed highest learning speed (Figure A.1.1).

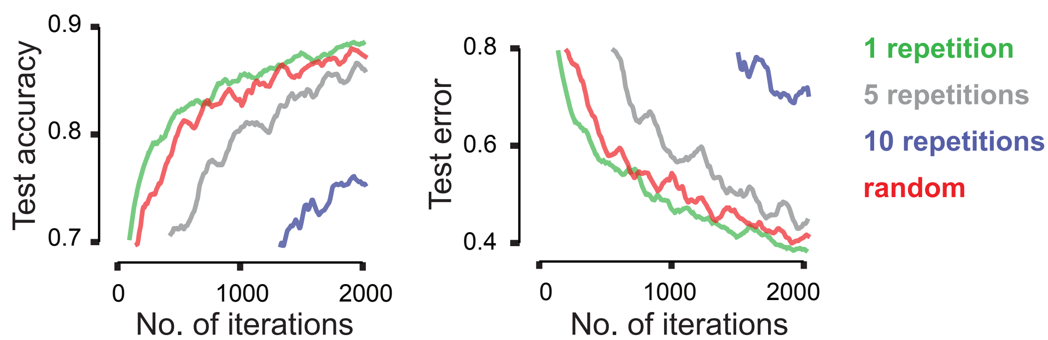

Figure A.1.1: Left: test accuracy for a memoryless classifier trained using CE loss on MNIST dataset. Right: test error (CE loss) for a memoryless classifier trained using CE loss on MNIST dataset. Hyperparameters were identical to the ones explained in section 3.1.

Initialization and Optimization Methods
We tested the model both with and without RMSprop optimization, along with Xavier initialization method (Tieleman and Hinton, 2012; Glorot and Bengio, 2010). When RMSprop was implemented, beta-1 and beta-2 were set to 0.9 and 0.99, respectively (Ruder, 2016).

Synthesized dataset with low category overlap
We synthesized a dataset with low category overlap consisting of 4 categories, each with 300 training items. Each item was a 1-by-16 vector. Different examples of a category were created by adding uniform noise to the template of the category (Figure A.1.2).

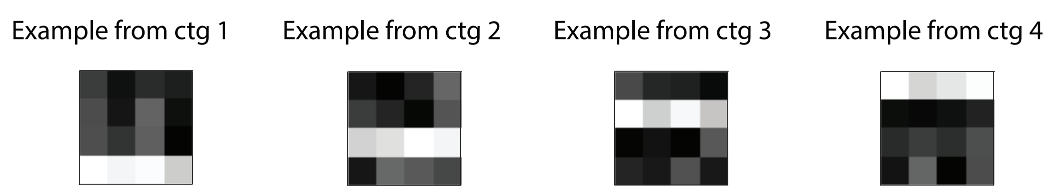

Figure A.1.2: Sample items from each of 4 categories in the synthetic dataset.

Category learning in neural networks with memory and gating for synthetic dataset
Figure A.1.3 shows effects of smoothness on neural network models equipped with leaky memory and gating for the synthetic dataset. Similar to the pattern observed in Figure 1.D, here we can see that in the network with leaky memory, higher levels of smoothness show better performance. Moreover, adding a gating mechanism enhanced learning such that all levels of smoothness surpassed minimum smoothness (1 repetition), as was observed in Figure 1.E (Figure A.1.3).

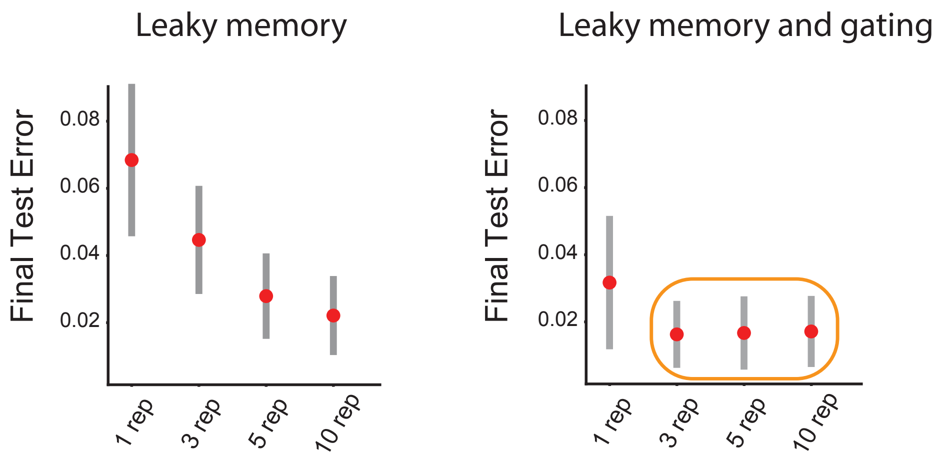

Figure A.1.3: The results of learning efficiency were generalized to synthetic dataset with low category overlap. Left: Test error (MSE loss) at the end of first training epoch with SGD on synthetic dataset, for network with leaky memory in internal representation. Right: The same as left, but for networks with leaky memory in internal representations and gating. Error bars show the mean and standard deviation of bootstrapping 10,000 times on 100 values achieved from 100 runs with different weight initialization.

How our approach differs from averaging in mini-batch training and momentum optimization
What we are doing here is different from mini-batch training and momentum optimization. In those methods, the smoothness is in the gradients (mini-batch training) and weight-updates (momentum), whereas here we are studying smoothness in the activation patterns.

Effects of smooth data on mini-batch training
We explored how smooth data affects learning when weights are being updated using mini-batch training. We used MNIST dataset and trained it with batches of size 16. Network dimension and other hyperparameters were identical to those used in incremental SGD. Our results showed that smoothness does not influence mini-batch training similar to SGD. Early in the training, minimum smoothness showed highest learning speed and higher levels of smoothness showed lower learning speed (Figure A.1.4). Whereas later in the training, another pattern was observed: the condition with the smoothness level equal to the batch size (e.g. 16 repetitions for batch of 16) showed highest learning efficiency compared to both lower levels of smoothness (e.g. 10 repetitions) and higher levels of smoothness (e.g. 24 repetitions) (Figure A.1.4).

One way to think about the observed results could be that in mini-batching, smoothness can happen at 2 levels: smoothness within a batch and smoothness across batches. It seems that early in the training, the condition with "minimum within-batch smoothness" has the highest learning speed. Minimum within-batch smoothness refers to the condition that has no smoothness inside a training batch. However, later in the training, the condition with minimum across-batch smoothness has the best learning speed. Minimum within-batch smoothness refers to the condition that has no smoothness inside a training batch, and minimum across-batch smoothness refers to the condition where each batch consists of items from only one category (e.g. 16 repetitions for batch of 16). This condition can be thought of as having minimum smoothness at the batch level.

Future work needs to further investigate how smooth data interact with mini-batch training.

A.2 Further details about testing the representations learned by brain-inspired architectures

Synthesizing simplified dataset with multiple levels of smoothness
We created training items with 3 features. Each feature consisted of 2 elements, forming a 3-by-2 item (Figure A.2.1). To form a training sequence with multiple levels of smoothness, we ordered

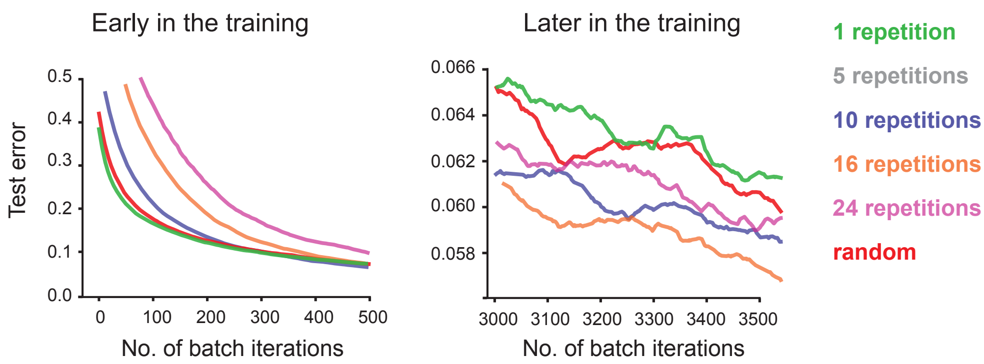

Figure A.1.4: Left: Test error (MSE loss) in different levels of smoothness in data, early in mini-batch training of MNIST dataset for classification. Right: The same as left, for later in the training, toward the end of first epoch.

training items such that different features varied at different scales: top row changed every item, representing a feature changing at a fast timescale; middle row changed every 3 items, representing a feature changing at a medium timescale; bottom row changed every 5 items, representing a feature changing at a slow timescale. Each element in each feature was the sum of an average feature-value plus random noise from a uniform distribution. The average feature-value was the same for all levels, therefore the only difference between levels of smoothness was the rate of change in the features. For creating the dataset, and designing and analyzing the models we used Numpy (Harris et al., 2020).

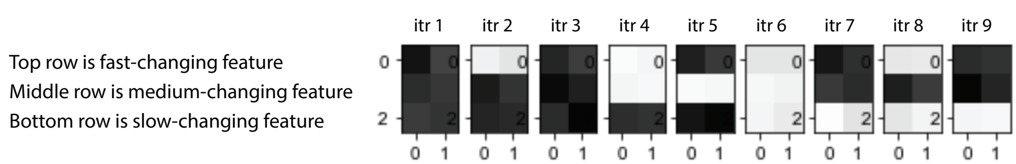

Figure A.2.1: First 9 training items. Each item is the input and the desired output at each iteration. Top, middle, and bottom row change every 1, 3, and 5 items, which results in 3 levels of smoothness.

Learning algorithm, optimization, and initialization

We used backpropagation with MSE loss, with RMSprop optimization method, and Xavier initialization (Tieleman and Hinton, 2012; Glorot and Bengio, 2010). We applied ReLU and Sigmoid as activation functions for hidden and output units, respectively.In RMSprop, the beta-1 and beta-2 were set to 0.9 and 0.99.

Hyperparameters

All 4 tested networks were 3-layer, fully connected autoencoders with (6, 3, 6) dimension. The learning rate was 0.01. For leaky memory in internal representations alpha in Eq.(1) was set to 0.5 (Figure 2).

Test error in unsupervised learning models

Before evaluating models' ability to "un-mix" timescales of the input, we first confirmed that all of the autoencoder (AE) models could learn to reconstruct the input. The two most efficient architectures were the multi-scale leaky AE with gating and the memoryless AE (Figure A.2.2).

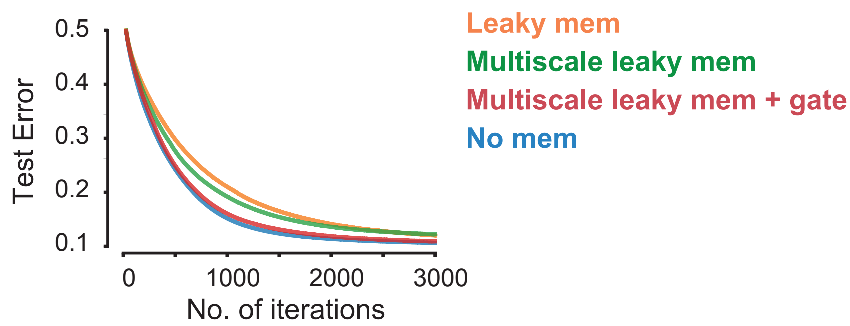

Figure A.2.2: Comparing reconstruction test error (MSE loss) during training for learning individual items across 4 different AE models. All the curves in this plot have been averaged over 20 runs with different initializations.