# Supplemental Material for the ICML Submission 'Understanding and Controlling Memory in Recurrent Neural Networks'

## 1. Delayed Template Matching Task

To examine the relevance of our results to other tasks, we considered a delayed template matching setup (also known as delayed match to sample). Here instead of introducing a trigger via an extra input channel, we introduce a second stimulus. When the second stimulus is introduced, the system is to report whether both stimuli belong to the same class or not. At all other times, the system should report a null output. We trained both architectures on the MNIST dataset, with the same minimal and maximal delays as the original task and an equal probability of both stimuli belonging or not belonging to the same class. As a result of the simpler nature of the template matching task compared to the classification task, both architectures were able to perform well without curricula training for nominal delays. We then repeated the analysis reported in Figure 4 of the main text. Figure A demonstrates here, similarly to the original task, the predictive nature of slow-point speeds. The faster the slow-point is, the less likely is the memory of the corresponding class to sustain for long delays.

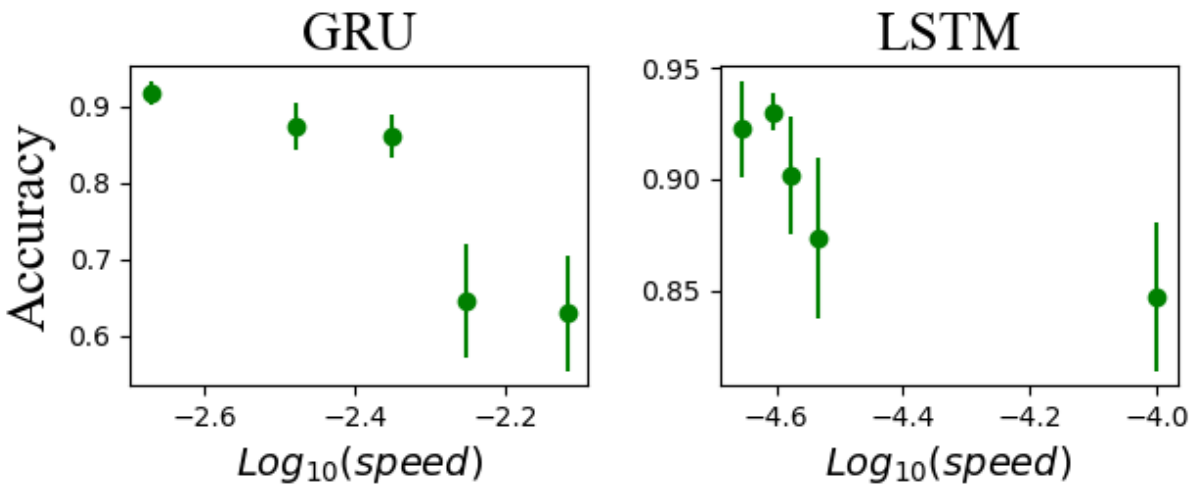


Figure A: Speed of slow-point and the accuracy of the corresponding class for a long delay on the delayed template matching task. Similarly to the main text Figure 4 both architectures exhibit a clear negative correlation between slow-point speed and the effectiveness for long delays. Note that the delayed template matching task was learn-able without any curricula, further generalizing our findings to naive training. Ten networks for each architecture were used.

## 2. LSTM Branching

Performing the branching analysis of Section 7 but for the LSTM architechure instead of GRU. Just as for GRU, Figure B shows that the details of slow-point branching affect performance. In the VoCu protocol, the introduction of new classes is accompanied by branching of an existing slow point. The classes associated with this spawning slow point are significantly adversely affected by this event, compared to other classes.

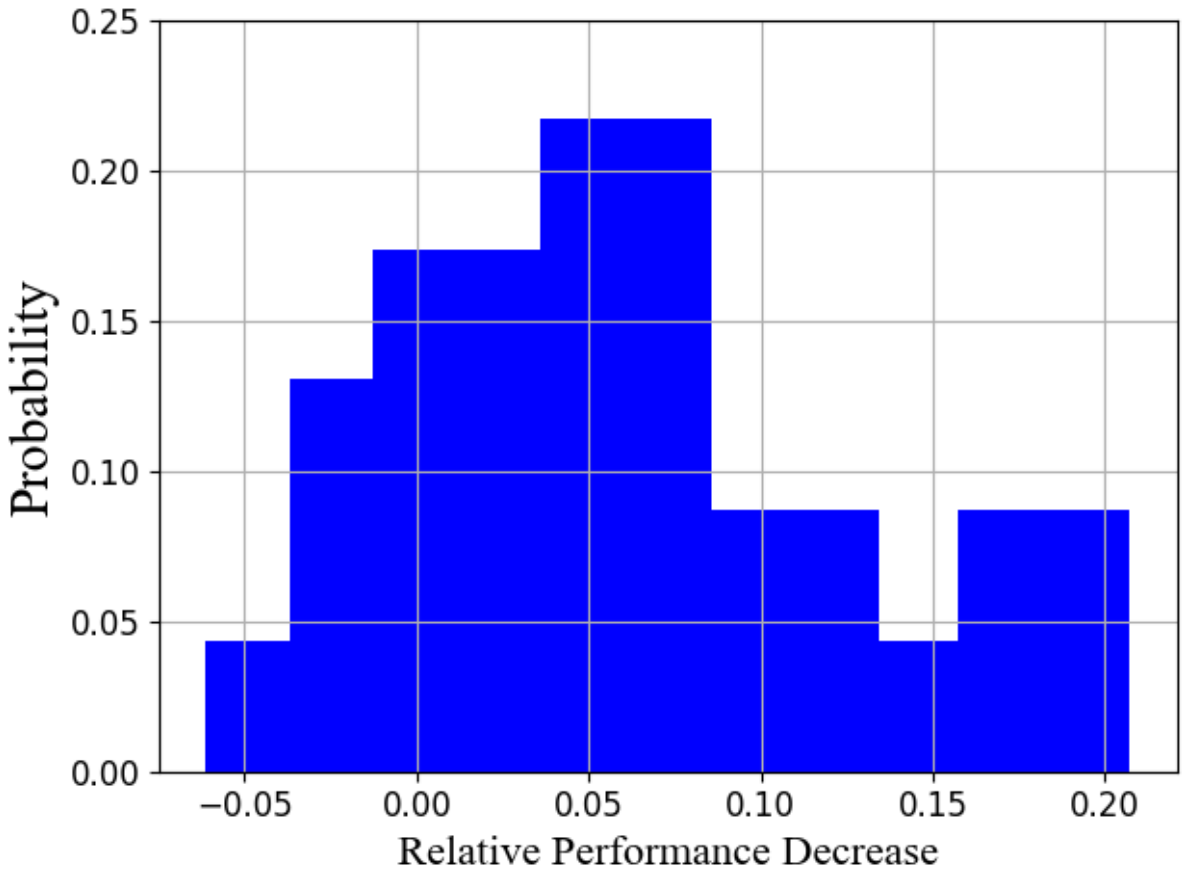


Figure B: Same as main text Figure 5E but for LSTM instead of GRU. Here as well, classes which split when new classes were spawned were more adversely affected compared to classes which did not experience a branching.

## 3. Regularization Figures

Results of Section 8 to all the data sets, training curricula and unit types are reported in Figure C. Following the trend of section 8, when regularizing the speed of the center-of-mass of each class or of the appropriate slow-point superior results were achieved for extended delays compared to when no regularization was applied ($\lambda \equiv 0$).

## 4. Validity of backtracking procedure during introductions of new classes in VoCu

Backtracking of slow point speeds around an event of introduction of new class reveals a spurious behavior of these speeds. We validated that even at these training epochs the tracking follows specific slow points and does not just capture random slow points in the hidden representation space. Fig. D depicts displacements $|\xi_k(\tau_i - 1) - \xi_k(\tau_i + 1)|$ of slow points associated with classes $1 \leq k \leq i$ back tracked along the step of introduction of a new class $c_i$, and compares them to distances to other slow points being tracked. It confirms that displacements of the points claimed to be tracked are indeed systematically lower than distances from other slow points under tracking. Importantly this is also true for the slow point $\xi_i$ which is assigned to a newly introduced class.

## 5. Accuracies on Test and Train sets for nominal task

Accuracies on the MNIST and CIFAR-10 train and test sets for the nominal task, for both GRU and LSTM architectures and the described curricula are reported in tables A,B,C,D.

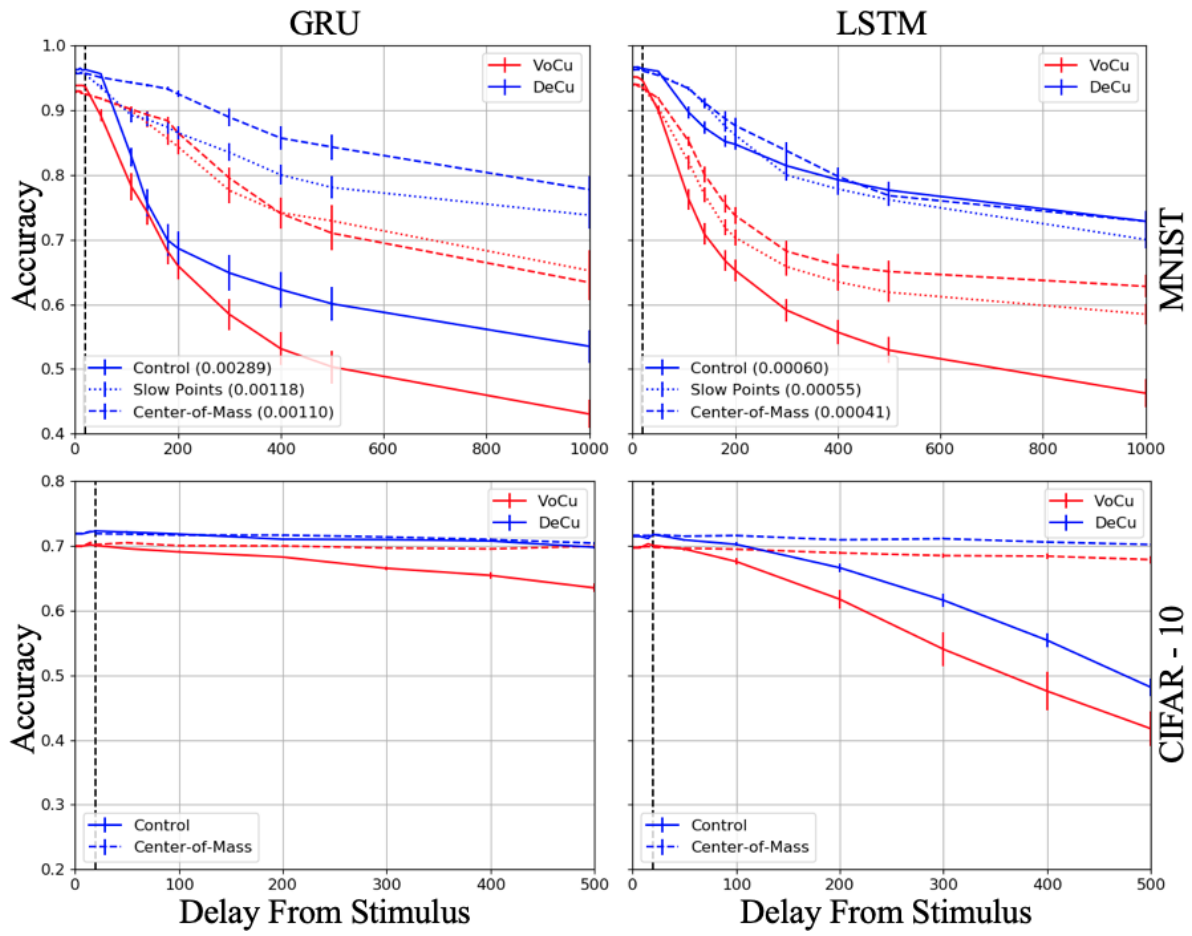


Figure C: Effect of speed regularization on the performance is demonstrated for all the test cases from Figure 6.

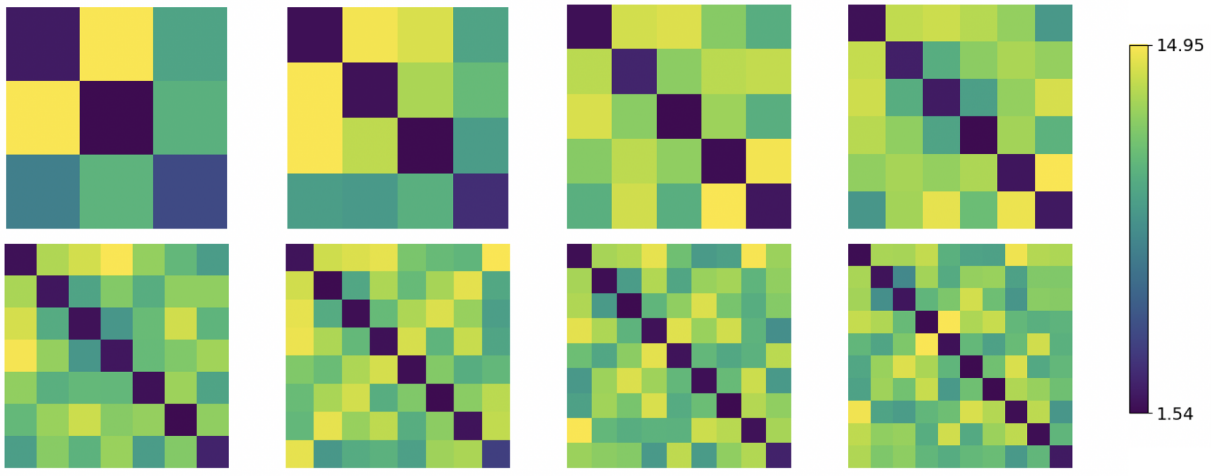


Figure D: Displacement of slow points along the training steps where new classes introduced is shown for all steps of a VoCu training example. Newly introduced class is always at the bottom-right corner.

Table A: MNIST - LSTM

| | | DeCu | VoCu |
|---|---|---|---|
| Training Set | Null | 100% | 100% |
| | Digits | $97.3 \pm 0.1\%$ | $95.2 \pm 0.3\%$ |
| Test Set | Null | 100% | 100% |
| | Digits | $97.3 \pm 0.1\%$ | $95.2 \pm 0.2\%$ |

Table B: MNIST - GRU

| | | DeCu | VoCu |
|---|---|---|---|
| Training Set | Null | 100% | 100% |
| | Digits | $97.5 \pm 0.2\%$ | $94.3 \pm 0.8\%$ |
| Test Set | Null | 100% | 100% |
| | Digits | $94.3 \pm 0.8\%$ | $94.3 \pm 0.8\%$ |

Table C: CIFAR - LSTM

| | | DeCu | VoCu |
|---|---|---|---|
| Training Set | Null | 100% | 100% |
| | Digits | $96.8 \pm 0.2\%$ | $97.1 \pm 0.2\%$ |
| Test Set | Null | 100% | 100% |
| | Digits | $71.2 \pm 0.5\%$ | $69.8 \pm 0.8\%$ |

Table D: CIFAR - GRU

| | | DeCu | VoCu |
|---|---|---|---|
| Training Set | Null | 100% | 100% |
| | Digits | $97.12 \pm 0.2\%$ | $97.0 \pm 0.7\%$ |
| Test Set | Null | 100% | 100% |
| | Digits | $72.0 \pm 0.3\%$ | $69.8 \pm 0.3\%$ |