A313 Cognitive Neurorobotics: Project Report
Investigating Adaptive Chunk-based Composition in RNNs

Student Name: Pranshu Malik
Student ID: 2401050
Date of Submission: 17 May 2025

Abstract
We investigated whether gated-recurrent unit (GRU) based recurrent neural networks (RNNs) can learn compositional structures, such as shared subsequences or "chunks", shared across different training examples. Specifically, we trained these RNNs to reproduce a small set of words written in cursive handwriting with shared subsequences at various positions (progressive similarities or differences) and then analyzed their learnt outputs and hidden activities to identify any chunk-based composition strategies qualitatively and quantitatively. The code and trained models for this project are available on `Github` and the results are also compiled in these `slides`.

Introduction and Methodology
We trained GRU-based RNNs on four cursively handwritten words (i.e. human action sequences) of four letters each: "well", "hell", "help", and "weld" (Figure 1). Each word contains shared subsequences (or "chunks") at different positions, such as "wel" or "hel" at the beginning, "el" in the middle, and "ell" at the end (Figure 2). Our objective was to determine if these shared chunks are encoded within the network's hidden states during sequence reproduction. Importantly, each trial began from an identical initial context (of zeroes) and the same starting position (also zero), with the one-hot encoded target-specifying vector provided as input at every timestep.

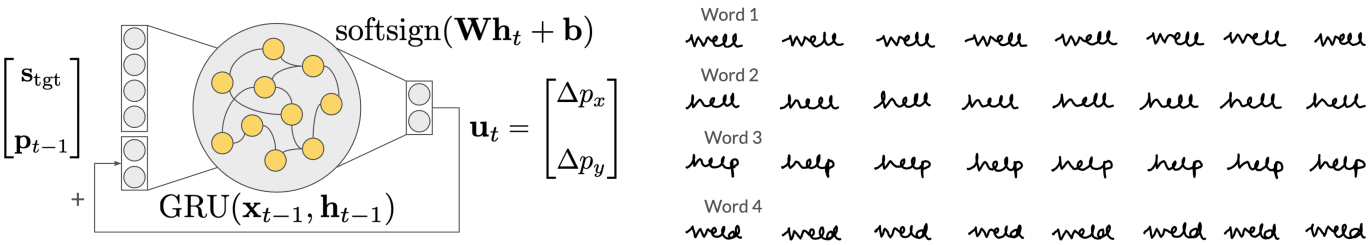

Figure 1: Model architecture and training data

Each sequence comprised $T = 74$ steps, excluding the initial position. Training datapoints were uniformly sampled along SVG paths representing the handwritten words. Alongside the one-hot encoded target vector ($\boldsymbol{s}_{\text{tgt}}$) representing executive intention, the network received the endpoint position ($\boldsymbol{p}_{t-1}$) as input and produced a change in position ($\boldsymbol{u}_t$) as output at each timestep, mimicking a sensorimotor task. This setup allowed easy "mixing" of targets as instructions to test for common elements and allowed silencing hidden units as well as principal components without any bias, due to the initial context being zero.

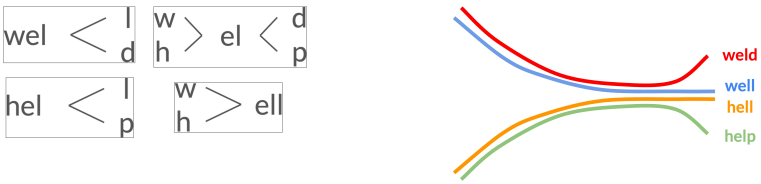

Figure 2: Some possible chunk-based compositions and progression of their similarities and differences

Training loss at each step was the squared distance between true sequence position and the network's output action applied to its position input, which was either the true previous global position (in teacher-forcing) or the cumulative sum of previous actions (in fine-tuning). Given the long prediction horizon, training was structured incrementally; we progressively increased the sequence length from 30 steps through 40, 50, 60, and finally to 74 steps. Concurrently, we adjusted the fine-tuning ratio from pure teacher-forcing (0) towards pure fine-tuning (1) in increments of 0.25. The training epochs were similarly scaled, beginning at 1000 epochs and increasing to 6000 epochs in the final stage. Note that for intermediate fine-tuning ratios (between 0 and 1), the choice of input (teacher-forcing or fine-tuning) at each timestep was probabilistically sampled from a Bernoulli distribution based

on that ratio and this was consistently accounted for during backpropagation through time. We formalize this below.

$$\boldsymbol{u}_t := \begin{bmatrix}\Delta p_x & \Delta p_y\end{bmatrix} = \text{Network}(\boldsymbol{x}_{t-1} \mid \Omega_{t-1}), \quad t \geq 1$$

$$\boldsymbol{x}_t := \begin{bmatrix}\boldsymbol{s}_{\text{tgt}} & \boldsymbol{p}_t\end{bmatrix}$$

$$\boldsymbol{p}_t = \begin{cases}\text{zero} & t = 0 \\ \hat{\boldsymbol{y}}_t & \text{if fine-tune and } t \geq 1 \\ \boldsymbol{y}_t & \text{otherwise for } t \geq 1\end{cases}$$

$$\hat{\boldsymbol{y}}_t = \begin{cases}\boldsymbol{u}_t + \boldsymbol{p}_{t-1} & \text{in general} \\ \sum\limits_{1 \leq \tau \leq t} \boldsymbol{u}_\tau & \text{reduced form in pure fine-tuning}\end{cases}$$

$$E_t = \|\boldsymbol{y}_t - (\boldsymbol{u}_t + \boldsymbol{p}_{t-1})\|^2, \quad t \geq 1$$

The accuracy of the network at any instant during training was estimated by calculating the mean loss (i.e. mean of deviation errors at each timestep) over the entire prediction length given a fixed set of parameters,

$$E_{\text{mean}} = \frac{1}{T} \sum_{1 \leq t \leq T} E_t(\Omega).$$

Results and Discussion

We first see the effect of model size on training performance when the network receives the cumulative sum of its outputs as feedback while generating the target sequences (Figure 3). In closed-loop generation, the model needed around a minimum of 16 hidden units to reach good task performance over 15000 training epochs. Qualitative similarities indicative of shared elements or patterns (i.e. chunks) are highlighted.

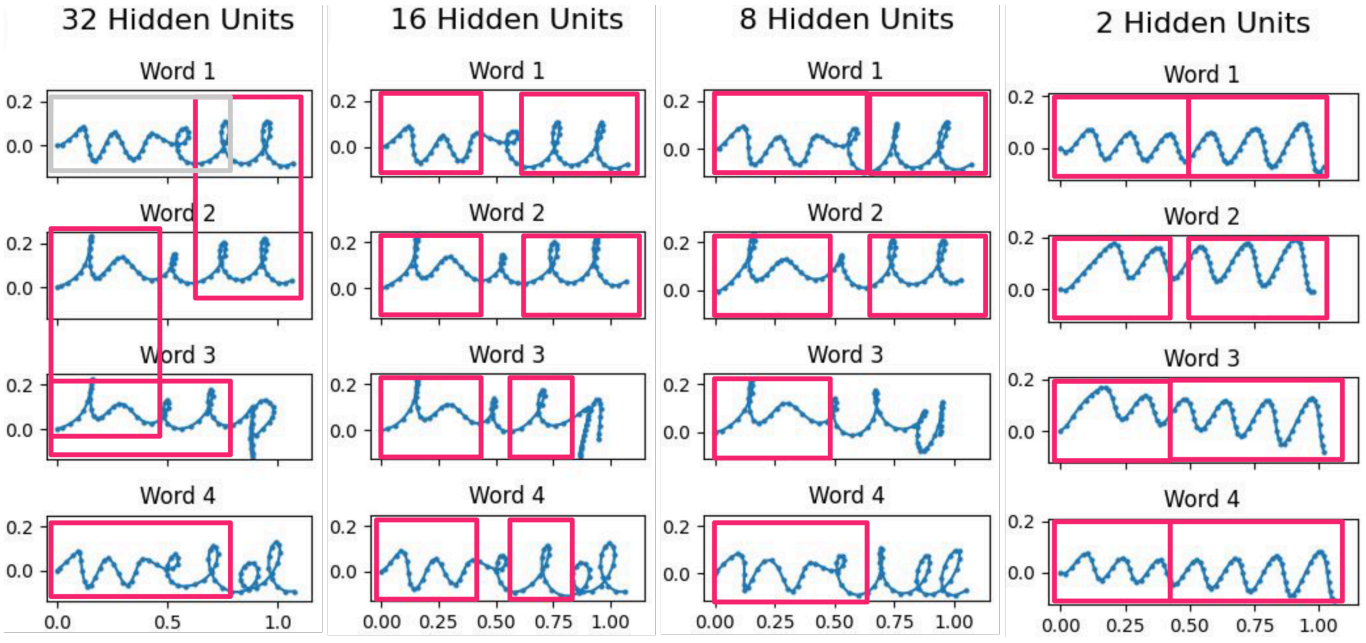

Figure 3: Feedback training performance

To understand the computation behind generation of these sequences, we examined the target-mixing effects and open-loop performance for each model to estimate the presence of any chunking (Figure 4). The idea behind target mixing was to see if mild corruption (15%-85%) or full competition (50%-50%) between two target patterns during closed-loop generation still preserves any dominant or shared pattern, which could be indicative of chunking. The figure shows that initial common elements between target patterns are preserved despite competition, and that the higher-dimensional model is robust to corruption by a more dissimilar target. Moreover, open-loop performance (with feedback held constant) shows that the models express baseline bias present in target patterns (i.e. 'w' words being lower than 'h' words due to starting from the same position) and primarily function as feedback controllers.

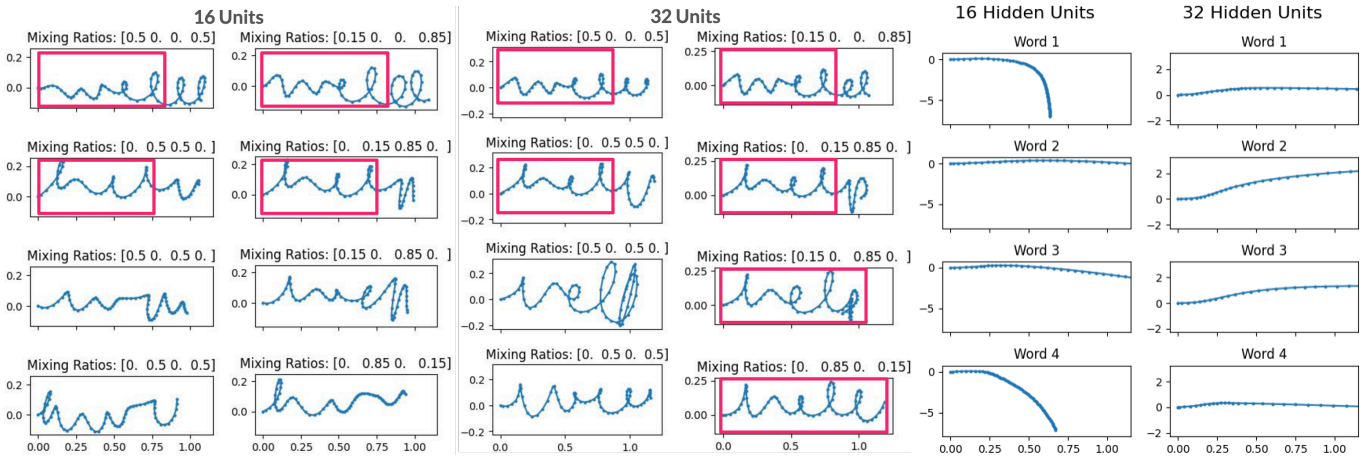

Figure 4: Target mixing and open-loop performance

Training in both Feedback and Feedforward Settings

In order to push the networks to learn more systematic elements intrinsic to the target sequences, rather than only learning "quick and dirty" feedback control tricks, we trained the models over an equal mixture of closed-loop (feedback) and open-loop (feedforward) trials using the same loss function and doubled the total number of training epochs. This approach allowed us to more explicitly test whether the models can learn to reuse dynamic components within the recurrent network when generating similar sequences, which also provides a better opportunity to study the underlying computational mechanisms. Particularly, by reducing the network's reliance on immediate feedback, we aimed to make the models transition from predominantly being feedback controllers towards being more balanced feedforward-feedback controllers. We expected this shift to prioritize learning shared structural elements or "chunks", such as 'hel', 'wel', and 'ell', as being adaptively composable learned actions.

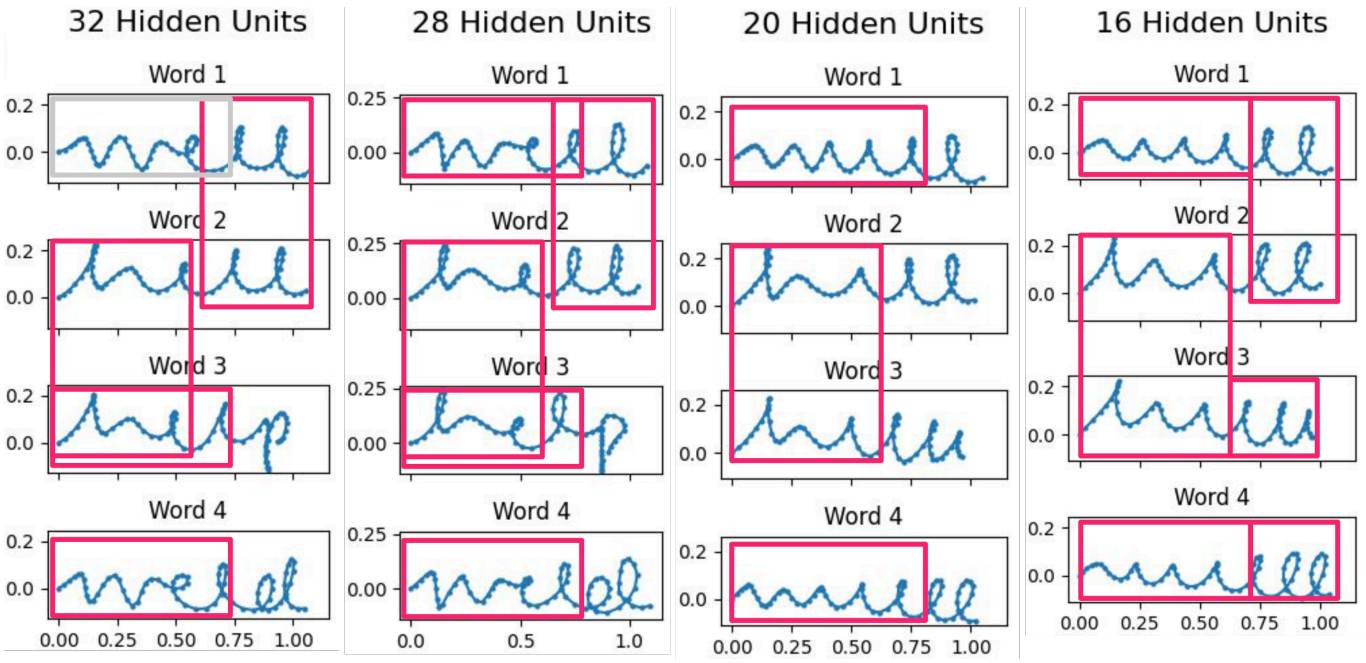

Figure 5: Feedback and feedforward training performance

The same kind of qualitative similarities (as Figure 3) suggestive of chunking are also apparent in feedback-feedforward training performance (Figure 5). However, achieving good performance in this more challenging setting necessitated higher-dimensional models, with $\gtrsim 32$ hidden units, to effectively embed the feature complexity in the four-word dataset during both closed- and open-loop generation. Similar chunking patterns are also visible and robustly preserved after mixing targets for the high-dimensional model (Figure 6). Finally, open-loop generation also indicated chunking, notably with 'weld' written similarly to 'well' but exhibiting drift reminiscent of a de-afferentiated patient writing without visual feedback (Smyth and Silvers 1987; Teasdale et al. 1993).

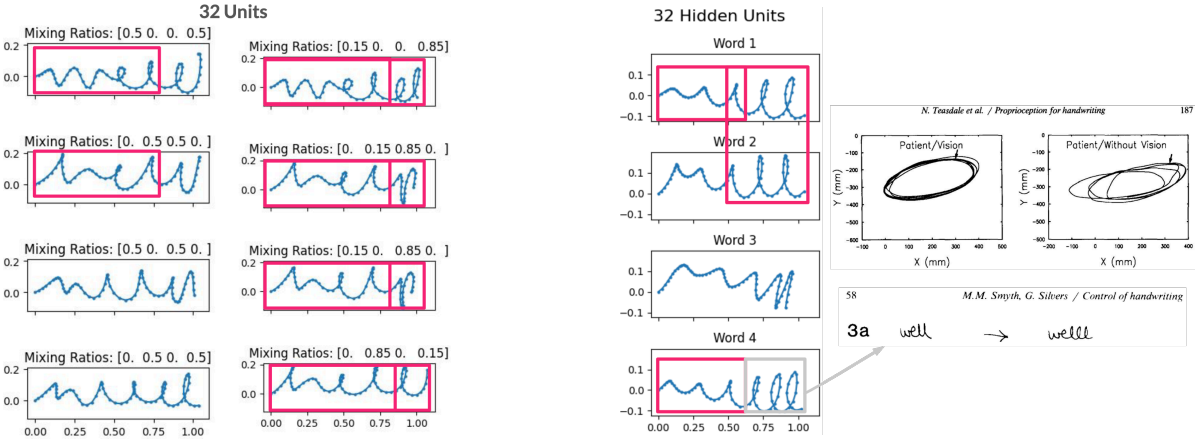

Figure 6: Target mixing and open-loop performance

Analyzing Hidden Activity of the Network

To understand the neural mechanisms underlying sequence generation in these models, we analyzed the hidden unit activity of the trained networks (e.g. selected units in Figure 7). Qualitatively, under feedback-only training, larger networks showed more systematic chunking-dependent features in raw hidden-unit activity, while smaller networks seemed to rely more on target- and state-dependent (feedback) strategies without any such apparent chunk-specific features. With feedback-feedforward training, some units in higher-dimensional networks exhibited temporal target-dependent similarities akin to adaptive chunking, whereas units in smaller networks displayed some chunk-dependent partitioning but with a weaker presence of temporal changes between those partitions, somewhat resembling the nature of patterns seen for larger networks trained in a feedback setting only.

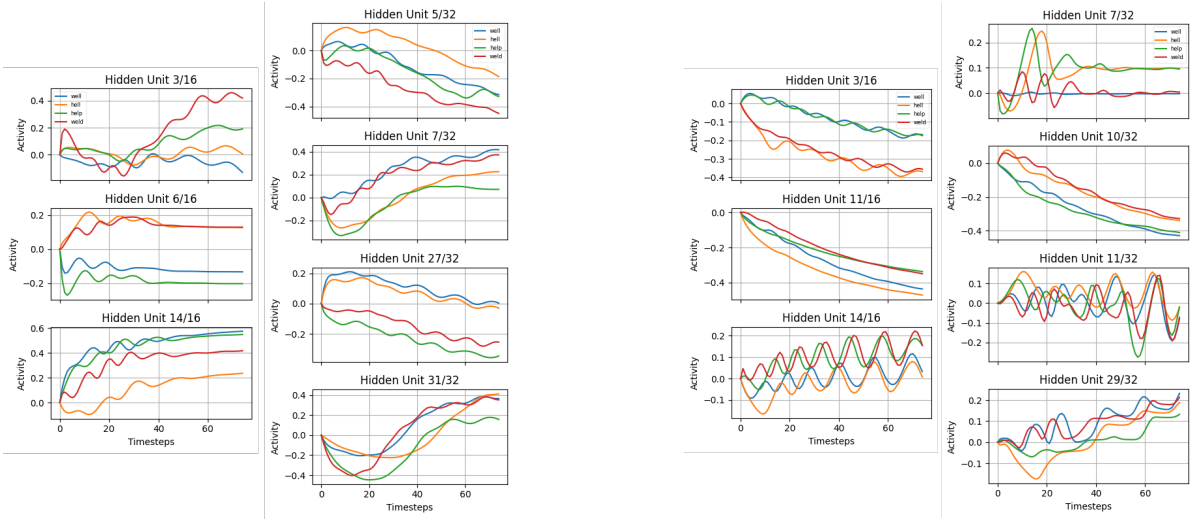

Figure 7: Hidden unit activities from feedback training (left) and feedback-feedforward training (right)

For a more objective analysis of hidden unit activity across time, target, and their interaction, we used demixed Principal Component Analysis (dPCA) as developed by Kobak et al. (2016). Distinct temporal phases (early, middle, late) in sequence generation were revealed by dPCA for models trained under feedback only, a feature less evident as such in the feedback-feedforward trained high-dimensional models (Figures 8-10, first row in dPCA plots). Target-dependent dPCs (middle row in dPCA plots) showed more systematic (chunk-dependent) similarities for higher-dimensional models across both training regimes, supporting the view that smaller networks act more as simpler feedback controllers. Time-target interaction dPCs (last row in dPCA plots) show no clear adaptively chunked components for feedback-only trained models, but there are stronger hints for it in high-dimensional models under feedback-feedforward training. To understand the functional roles of these dPCs, we also examined the effects of silencing them. Smaller networks were completely disrupted by silencing even the least dominant target-dPC (Figure 8). In contrast, silencing time-target-dPCs in higher-dimensional models had similar impacts on ‘w’ and ‘h’ words (Figure 9), or affected the generation of “ell” in “well” and “hell” for both target and time-target dPCs (Figure 10), providing clear evidence for chunk-dependent sequence generation.

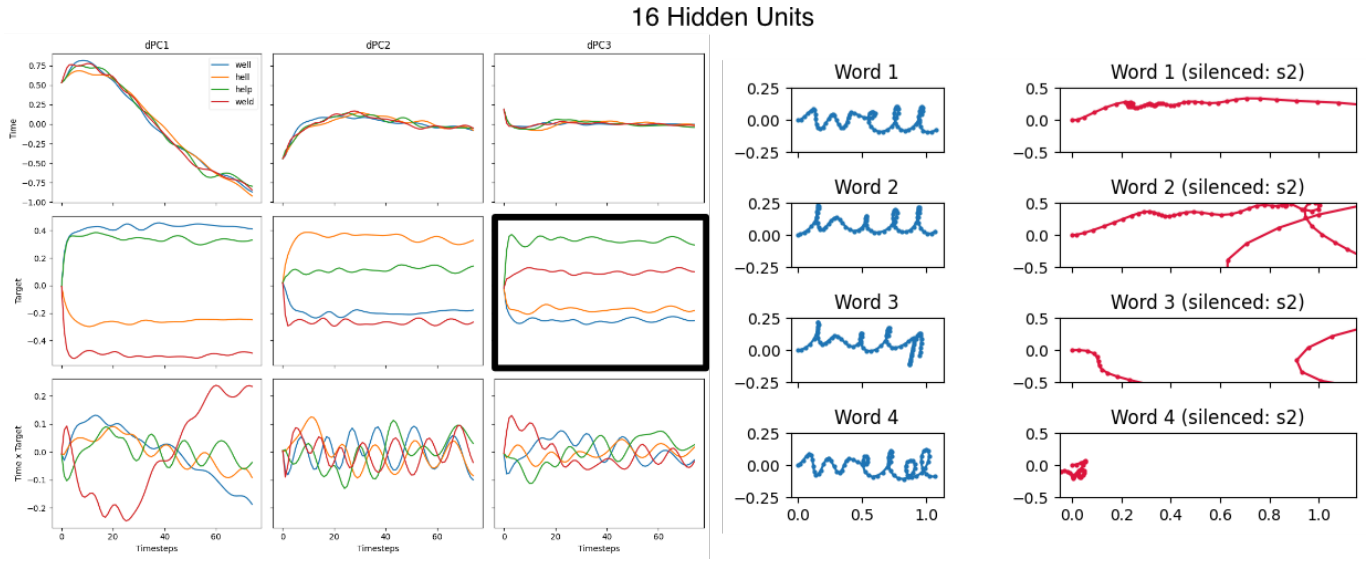

Figure 8: dPCA analysis for model with 16 hidden units under feedback training

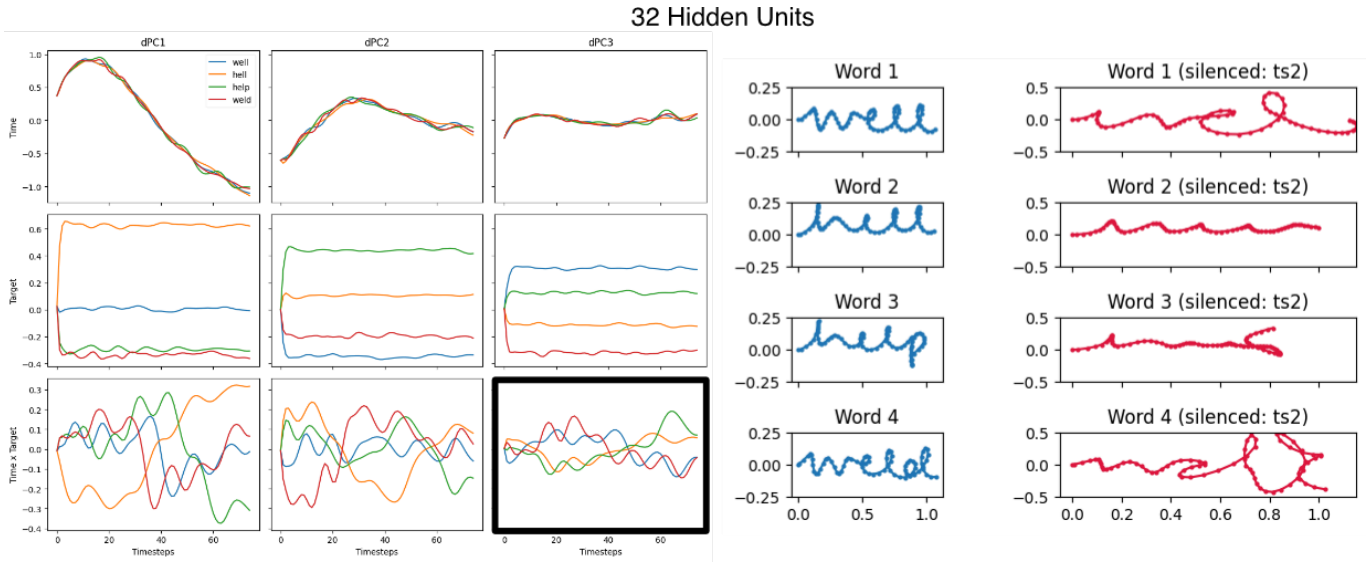

Figure 9: dPCA analysis for model with 32 hidden units under feedback training

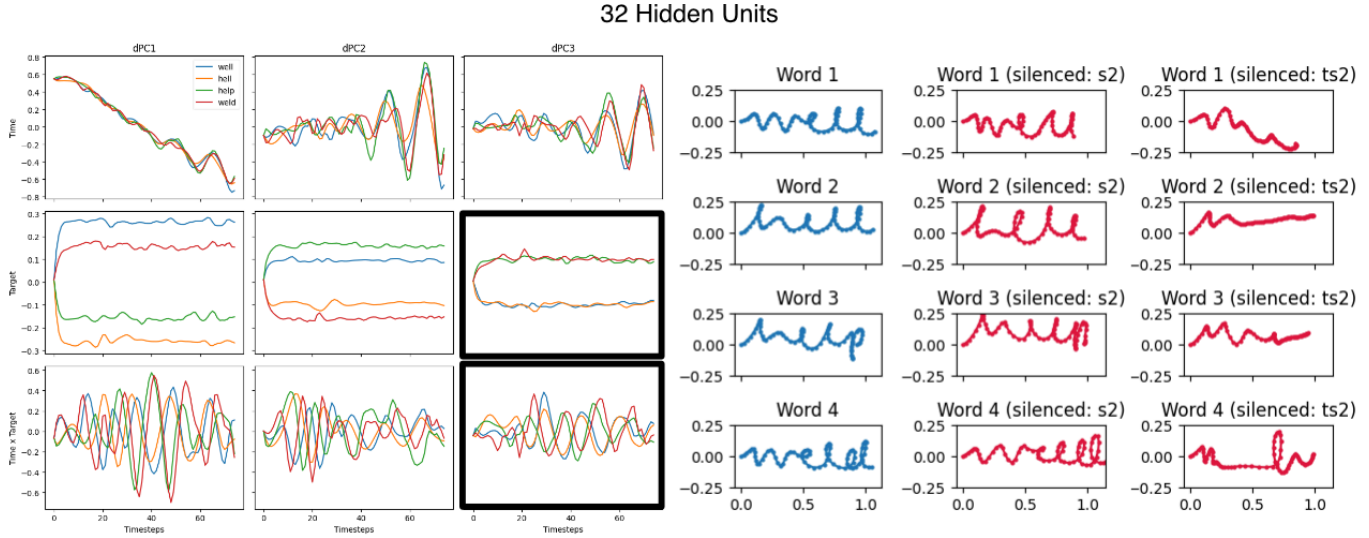

Figure 10: dPCA analysis for model with 32 hidden units under feedback-feedforward training

Further explorations to probe and understand neural computations in these RNNs include:

- Looking at individual hidden unit contributions by silencing them directly.
- Finding fixed points in the network's hidden activity (for example using this `toolbox`) to further understand how the computations evolve in time and if they are repurposed for different targets (see Sussillo and Barak 2013).
- Plotting the evolution of output phase space subject to the corresponding hidden context, to qualitatively see and contrast how position and target feedback shape the output dynamics. If this is done over hidden activity with respect to input position feedback, we can also see how the network partitions and/or reuses attractors in its latent dimensions (see Radhakrishnan et al. 2020).
- Silencing dPCs without introducing a compounding effect over time by instead seeing its effect independently at each timestep by allowing the ideal hidden states to evolve and only silencing the projection on that particular dPC while producing the output at each step.

It is also possible for better generalization (i.e. reuse) to occur if target words are on the same global baseline position. Currently, the words starting with 'w' are shifted down relative to words starting with 'h' due to the initial stroke having different heights. This will require shifting the input data accordingly (during pre-processing), and the networks can be trained in a re-run of the code notebook without much further changes. However, this is not a feature that is generally true for sequences in the real world, and such situations can be better handled with minor additions to the network architecture, and so we skipped it in this project.

Training with Delayed Feedback

To encourage the formation of internal representations that persist longer in time, are reused across multiple targets, and are less dependent on immediate feedback, we introduced a delay in the feedback provided to the network. This forced the models to simulate similar internal feedback for shared segments across target words, akin to open-loop trials but now within closed-loop operation as well. We hypothesized that this would promote adaptive chunking in the true sense (i.e. time-evolving), rather than simply using feedback to steer target-specific temporal bases to give rise to "chunks".

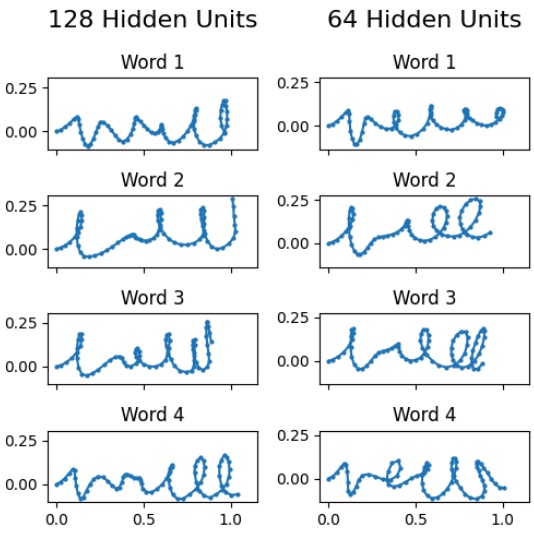

Figure 11: Delayed feedback training performance

As evident in Figure 11, even high-dimensional models (with 128 units) struggled to compensate for a delay of 10 time steps and produce legible output. This suggests they were unable to generate effective internal feedback within their capacity. It is plausible that networks with even higher dimensionality are needed to adequately embed the computations necessary for learning adaptive chunks across all target words.

Conclusion

Our investigation suggests the presence of compositionality in the trained RNNs, although not entirely of a "natural" kind. We observed indications of quasi-adaptive chunking with feedback-dependent steering of target-specific bases in feedback-only trained models. However, direct evidence for invariant and chunk-dependent shared representations evolving in time was limited and only hinted at in the feedback-feedforward trained models.

Limitations and Future Directions

It is possible that the relatively small training set of four words allowed the networks to primarily learn target-specific operations rather than necessitating robust chunk-based composition. Expanding the training set with words like "welp" and "held" could better balance pattern and target complexity, potentially encouraging more sophisticated learning strategies. Furthermore, the use of handwritten target sequences introduces the possibility of systematic biases inherent in the dataset (e.g. target-specific spread or contraction of strokes). We could explore automated or more controlled dataset generation. It is also plausible that architectural constraints, such as a hierarchical structure, might be necessary to elicit more apparent adaptive chunking. Additionally, introducing control constraints by having the network directly control a simulated arm-like appendage for "writing" (e.g. as in the `motornet` toolbox by Codol et al. 2024) could promote more naturalistic representations. Finally, we note a potential issue with the training data: "weld" and "help" had slightly longer path lengths compared to "well" and "hell". Our uniform sampling of 75 points likely resulted in longer substrokes for these words, which could have hindered the network's ability to learn chunk-based composition effectively. This limitation could be addressed in future work by using synthetic generation or uniform length-based sampling of the target words.

References

Codol, Olivier, Jonathan A Michaels, Mehrdad Kashefi, J Andrew Pruszynski, and Paul L Gribble. 2024. "Motornet, A Python Toolbox for Controlling Differentiable Biomechanical Effectors with Artificial Neural Networks". Edited by Juan Alvaro Gallego and Tamar R Makin. *Elife* 12 (July):RP88591. https://doi.org/10.7554/eLife.88591.

Kobak, Dmitry, Wieland Brendel, Christos Constantinidis, Claudia E Feierstein, Adam Kepecs, Zachary F Mainen, Xue-Lian Qi, Ranulfo Romo, Naoshige Uchida, and Christian K Machens. 2016. "Demixed Principal Component Analysis of Neural Population Data". Edited by Mark CW van Rossum. *Elife* 5 (April):e10989. https://doi.org/10.7554/eLife.10989.

Radhakrishnan, Adityanarayanan, Mikhail Belkin, and Caroline Uhler. 2020. "Overparameterized Neural Networks Implement Associative Memory". *Proc. Natl. Acad. Sci. U. S. A.* 117 (44): 27162–70.

Smyth, Mary M., and Gil Silvers. 1987. "Functions of Vision in the Control of Handwriting". *Acta Psychologica* 65 (1): 47–64. https://doi.org/https://doi.org/10.1016/0001-6918(87)90046-1.

Sussillo, David, and Omri Barak. 2013. "Opening the Black Box: Low-Dimensional Dynamics in High-Dimensional Recurrent Neural Networks". *Neural Comput.* 25 (3): 626–49.

Teasdale, N., R. Forget, C. Bard, J. Paillard, M. Fleury, and Y. Lamarre. 1993. "The Role of Proprioceptive Information for the Production of Isometric Forces and for Handwriting Tasks". *Acta Psychologica* 82 (1): 179–91. https://doi.org/https://doi.org/10.1016/0001-6918(93)90011-F.