# A Spiking Neural Model of Attention Effects in Memory

**Marshall L. Mykietyshyn (mmykiety@edu.uwaterloo.ca)**
**Terrence C. Stewart (tcstewar@uwaterloo.ca)**
Systems Design Engineering, University of Waterloo
200 University Avenue West, Waterloo, ON, N2L 3G1, Canada

**Abstract**

We present a spiking neuron model of attention-driven memory, where participants use a cue to indicate whether a word on a list is to be remembered or not. This model is fit to individual differences on mean behavioural data and produces a good match in terms of variance of performance on a recognition task, but not on a recall task. Neural activity patterns during the memorization parts of the task are also well-matched, but not during the time between when the attention cue is presented and when the word itself is presented. We believe this indicates mechanisms are involved in the recall task which are not considered as part of the current model.

**Keywords:** attention; memory; Neural Engineering Framework; LIF neurons

## Introduction

The overall goal of this work is to produce a neural-level explanation of psychological phenomena: in this case, the ability to control (via attention) what items on a word list are remembered, and which ones are not. In particular, we are interested in how low-level effects (such as how well neurons can represent, store, and transform information) can give explanations for how tasks are performed, and how people differ.

In an experiment to "isolate the neural mechanisms of attention that lead to improved memory formation," Wittig et al. (2018) visually presented words for one second, followed by a five second delay between words. Subjects were instructed to remember words that were preceded (cued) by a row of asterisks, with no instruction given regarding uncued words. Once the entire list was presented, subjects performed distraction tasks (simple arithmetic problems) for 20 seconds to suppress sub-vocal rehearsal and mitigate recency effects. Then the subjects memory of the word list was assessed using recognition and recall tests (John H. Wittig, Jang, Cocjin, Inati, & Zaghloul, 2018). This process is shown in Figure 1.

In the recognition (seen/unseen) test, subjects were shown a second list of words (test list) and asked to identify which words came from the list originally presented during the task (task list). The test list consisted of a mixture of cued words and uncued words from the task list, in addition to words that were not part of the task list (foil words). The recall test required subjects to verbally recite as many of the cued words as possible.

There were two recognition test criteria used to determine whether sessions would be used in the analysis 1) there must be a significant difference in recognition rates between cued and uncued words 2) there must be a significant difference in recognition rates between uncued and foil words. Significance was determined using a chi-squared contingency table (we assumed the criterion was $p<0.05$, although this is not stated in the paper). While training subjects to perform the task, the task difficulty was calibrated for each subject. Task difficulty was set primarily by altering the length of the task list such that subjects would meet both stated criteria. Other parameters (e.g. fraction of cued words) were altered if changing the list length was insufficient to meet the criteria; however, these secondary adjustments were not considered in the current work.

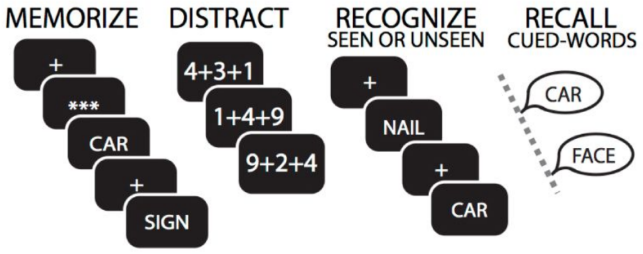


Figure 1: Task description, from (John H. Wittig et al., 2018).

Adjusting the task difficulty allowed the testers to "collect isoperformance data across participants who showed a wide range of natural aptitude for the task 1. Overall, 71/90 experimental sessions met both criteria, with training sessions excluded from the analysis. The distribution of list lengths is shown in Figure 2.

## Representation

The goal of this work is to build a computational model of the memory aspects of this task using spiking Leaky Integrate-and-Fire (LIF) neurons. We use the Neural Engineering Framework (Eliasmith & Anderson, 2003) to find the connection weights between these neurons such that a) each group of neurons forms a distributed representation of a vector and b) each set of connections between groups of neurons approximates some desired function on those vectors. The neurons themselves have randomly chosen properties such as maximum firing rates, tuning curves, and preferred stimuli (a.k.a. encoders) to give a realistic heterogeneous distribution

In particular, if a population of neurons is to represent the vector $\mathbf{x}$, then each neuron $i$ has an *encoder* $\mathbf{e}_i$ which is the value of $x$ for which it most strongly fires. If the neuron has a randomly chosen gain $\alpha_i$ and bias $\beta_i$ then the total current flowing into the neuron would ideally be $\alpha_i \mathbf{e}_i \cdot \mathbf{x} + \beta_i$. This will cause the population of neurons to have a different firing pattern for every value of $\mathbf{x}$.

If one group of neurons represents $\mathbf{x}$ and another represents $\mathbf{y}$ and we want $\mathbf{y}$ to be some function of $\mathbf{x}$ (i.e. $\mathbf{y} = f(\mathbf{x})$), then


Figure 2: Distribution of list lengths among the 18 subjects, from (John H. Wittig et al., 2018).

we can form connections between the two groups of neurons. We solve for the weights $\omega_{ij}$ between neuron $i$ in the first population and $j$ in the second population such that the spiking activity $a_i$ in the first population when representing $x$ will lead to the corresponding current flowing into the second population to represent $f(x)$. In other words, we want $\sum_i a_i \omega_{ij} = \alpha_j \mathbf{e}_j \cdot f(\mathbf{x})$. Given this formulation, the connection weights $\omega_{ij}$ for any given function $f(\mathbf{x})$ can be found using least-squares minimization. We use the software package Nengo (Bekolay et al., 2014) to automate this process and run the resulting model.

While the above method can be used to create biologically plausible implementations of any computation, the neurons never perfectly approximate that computation. For example, if a group of neurons is connected back to itself with connection weights approximating the function $\mathbf{y} = \mathbf{x}$, then this would ideally be a perfect memory, storing the information $\mathbf{x}$ over time without change. However, the neural activity will, in practice, gradually change, leading to drift in the value $\mathbf{x}$ that is being represented. The purpose of this paper is to explore how that sort of low-level implementation detail affects the performance of an attention-driven memory system.

Because this is our goal, here we only present a model of the attention-driven memory. We do not include here a model of the visual recognition of cues and words, or the cognitive control needed to perform the task itself, as these aspects have been previously modelled using the Neural Engineering Framework (Eliasmith et al., 2012). We also do not model the decision-making process required to decide whether or not a particular word was in the remembered list. Instead, we directly decode out the vector $\mathbf{x}$ in the memory (also via least-squares minimization) and compare it to the vectors for different words using the dot-product to measure similarity. This was done for simplicity, and is equivalent to (but less noisy than) more detailed decision-making models that have previously been published (Sharma, Komer, Stewart, & Eliasmith, 2016; Hurzook, Trujillo, & Eliasmith, 2013).

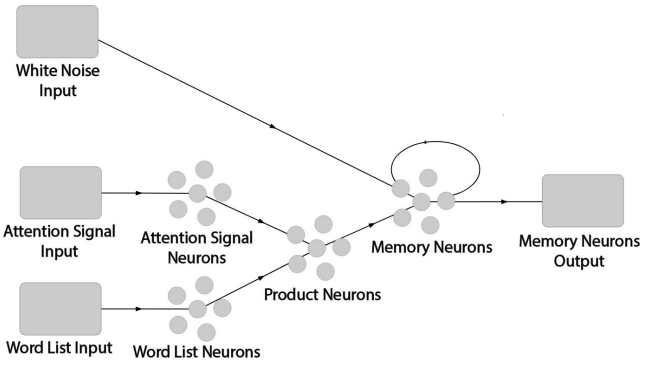


Figure 3: The spiking model of attention-driven memory.

## Model

To model the memory aspects of this task, we use a group of 51,200 neurons representing a 512-dimensional vector (*Memory Neurons*). These neurons are recurrently connected to compute the function $\mathbf{y} = \mathbf{x}$. This means that, in the absence of any input, the neurons will maintain their firing pattern, creating a memory (mathematically, it will compute the integral of its input). The presented word is represented using another group of 51,200 neurons, representing a 512-dimensional vector (*Word List Neurons*). The effect of the cue is represented with an attention signal: 100 neurons representing a scalar value of how much attention to pay to the current word (*Attention Signal Neurons*. The 102,400 *Product Neurons* multiply the attention value by the word vector, sending the result into the memory.

To model the task, we present as input the (randomly chosen) vector for the current word, and a scalar value of how much attention to pay to that word. This will be larger for cued words than for uncued words, but the exact values are fit to account for individual differences between subjects. We apply varying amounts of white noise to the memory, also to account for individual differences. An illustration of this model can be seen in Figure 3.

The word list is represented by N randomly chosen 512-dimensional normalized vectors, where N is the length of the list. The attention signal is either high (cued words) or low (uncued words) for the one second the word is presented, and zero for the five second delay between words. When the attention signal is zero, the memory neurons are only affected by the white noise input and the inherent error involved in approximating a perfect memory using spiking neurons. For all simulations, exactly half the words in the list are cued, mirroring the original experiment conditions considered here.

The memory vector acts as an information compass, where the direction it points in the 512-dimension space indicates what information it represents. As the product of the inputs accumulates in memory, the memory vector gradually turns toward the direction of the word vector. How far the memory vector turns depends on the magnitude of the input (i.e. whether the attention signal is high or low), how long the

word is presented for (one second in all cases), and how much information has previously been imprinted on the memory space. These effects can be seen in Figure 4. In all cases, we plot the dot product (i.e. the cosine similarity) of the vector decoded from memory and the ideal (randomly chosen) vector for each word.

The subjects memory at the time of testing is represented by the decoded memory vector at the end of the simulation. The 20 seconds of distraction tasks are represented by a 20 second period of zero input, during which no mechanism is applied to simulate rehearsal or other memory-enhancing effects. Although the recognition test was performed before the recall test in the experiment, for this work we did not consider how to model this interaction, which may be addressed in future work.

Since the model's memory vector represents the history of words imprinted on it, how strongly those words are held in memory can be calculated as the dot product of the original word vectors and the memory vector. Henceforth, we refer to this as cosine similarity. In the compass analogy, the magnitude of similarity indicates to what degree the memory and word vectors are pointing in the same direction. Cosine similarity can result in a value between 0 and 1, with 1 being perfect alignment and 0 being perfectly orthogonal. Since we are not yet building a full model of the decision-making process for extracting information out of memory, here we simply choose a threshold (different for each individual) for this similarity value.

Simulating the recognition and recall tests involved creating cosine similarity threshold values above which a word is represented strongly enough in memory to be considered "recognized" (seen) or to be "recalled". These were taken as separate threshold values, as it was assumed that the mental process for recognizing a word is different than for recalling one. In order to establish threshold values that would reflect the experimental responses, experimental data was used to determine thresholds for each of the categories "seen cued", "seen uncued", "seen foil", and "recalled". Thresholds were calculated using the experimental correct response rates, which can be seen in Figure 5 below. For each category, the value above which the proper percentile of word cosine similarities lay was taken as the threshold. For example, in the experimental results, approximately 90% of cued words were correctly identified in the recognition test. Therefore, the "seen cued" threshold was calculated as the value above which 90% of cued words cosine similarities lay, see Figure 4. The calculations were performed over 20 sessions to reduce sample size error, and account for the volatility individual sessions.

Note that foil words are a separate randomly generated list of word vectors that are never presented to memory. They represent words that are not part of the task list, but are shown during the recognition tests. The rate of foil words recognized (seen) is the false-alarm rate of the test. Even though the words are never presented, and therefore are never imprinted

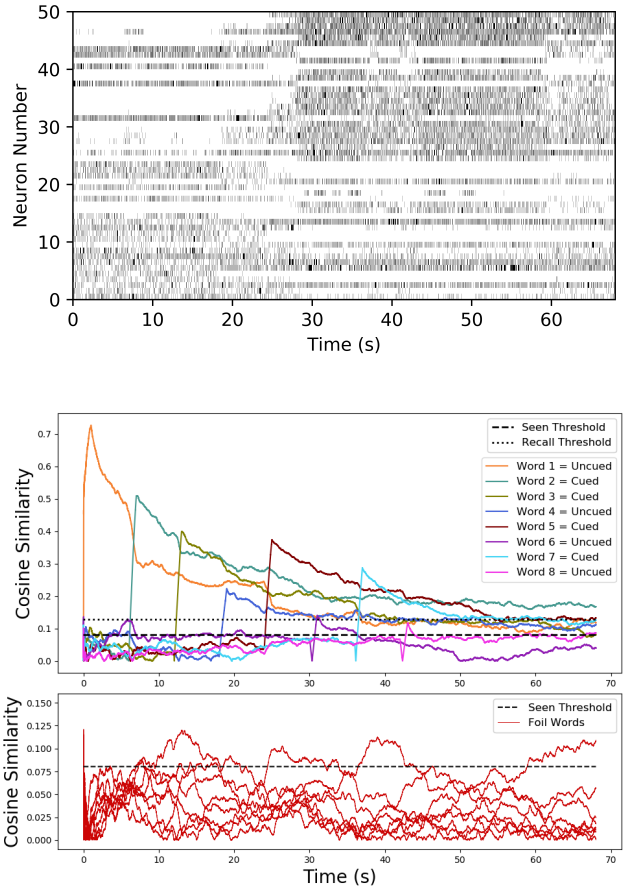


Figure 4: (Top) Spiking output from 50 randomly chosen Memory Neurons. This is the neural activity from which we decode the memory vector used for the other two parts of this figure. (Middle) Strength of task list words in memory, represented by the normalized cosine similarity of the memory vector with each studied word at each point in time. The calculated thresholds for the recognition and recall tests are shown for this subject (see below). (Bottom) Strength of foil words in memory. No foil words are imprinted on memory during simulation, so any similarity is simply due to the random foil word vectors existing in the same vector space as the memory vector. In this case, one foil word was incorrectly identified as being seen during the task. The "Seen Word Threshold" has the same value in the upper and lower plots.

on memory, their cosine similarity values are non-zero, since they exist in the same 512-dimension vector space as memory. This allows us to calculate a false-alarm rate for the simulation, representing random chance, which is analogous to the foil word recognition rate in the experiment.

Since the recognition test involved three different categories, three separate recognition thresholds were calculated. However, in order for a word in memory to be recognized, it is irrelevant which category that word comes from. Therefore, model parameters were adjusted such that the recognition thresholds for the three seen categories were close enough that using their average would produce results similar to the individual thresholds. This allowed us to calculate a single recognition threshold representing all "seen" categories. It was found that various combinations of parameters could produce approximately overlapping recognition thresholds for the three categories. This led to the creation of individual parameter profiles for each participant, examples of which can be seen in Table 1.

Table 1: Subject Profile Examples.

| Tuned Parameter | Sub #1 (N = 8) | Sub #13 (N = 16) | Sub #15 (N = 24) |
|---|---|---|---|
| Cued (High) Attention | 0.7 | 0.9 | 0.9 |
| Uncued (Low) Attention | 0.4 | 0.5 | 0.5 |
| White Noise | 0.01 | 0.008 | 0.003 |
| Simulation Output | | | |
| "Seen Cued" Threshold | 0.081 | 0.050 | 0.058 |
| "Seen Uncued" Threshold | 0.087 | 0.060 | 0.059 |
| "Seen Foil" Threshold | 0.073 | 0.074 | 0.071 |
| Average Seen Threshold | 0.080 | 0.061 | 0.063 |
| Recall Threshold | 0.128 | 0.106 | 0.106 |

## Tuning for Individual Differences in Task Difficulty

The original experiment used 18 subjects, whose corresponding word list lengths are shown in Figure 2. For the simulation, we created 18 subject profiles, tuned for list lengths matching the distribution of the original subjects. Tuning involved adjusting parameters such that the three "seen" category thresholds were similar in four out of five simulations. The parameters manipulated were the attention signal high/low values and the level of white noise added to memory. Once the subject profiles were determined, 30 simulations were run for each profile. The first 20 simulations were used to calculated recognition and recall cosine similarity thresholds, and the final 10 simulations were used as test sessions, upon which the analysis is performed.

White noise negatively affects the memory neurons ability to hold a value over time. Without the white noise, the "seen cued" and "seen uncued" thresholds were much higher than the "seen foil" threshold, which would have caused foil word recognition rates of near zero. Thus, adding this noise was necessary to accurately reflect the experimental responses. Another option for reducing the stability of the memory would have been to adjust the number of neurons. However, using white noise produced a much smoother effect on performance, making it easier to find parameter settings which matched particular subjects.

The experimental criteria described in the Task Description section were used to determine whether sessions were valid for analysis. Significance was calculated using a chi-squared contingency table, as described in the experimental methodology, using $p<0.05$.

It was found that subjects tuned for list lengths of four words resulted in a large majority of failed sessions (5/30 met the recognition test criteria). In a four word task list, two words are cued and two are uncued. Therefore, only two data points are available for calculating the "seen cued" and "seen uncued" thresholds. This made individual session thresholds for four word task lists quite volatile, with a large variance across the sessions used to calculate the average threshold. Consequently, the average thresholds for these subject profiles were not properly representative of the data, resulting in a high number of failed tests. Furthermore, the tests that did pass the statistical criteria, did not provide recognition rates in the expected ranges. Therefore, subject profiles with four-word task lists were removed, leaving 15 subject profiles for analysis. Additionally, Figure 2 shows only one subject with a task list length of zero. This subject was replaced with one where N = 16.

Since the thresholds were based upon the experimental data, and the parameters were tuned such that the seen thresholds would overlap, there was some concern about over-tuning the model. This is investigated in the Model Exploration section below.

## Stages of Analysis

The model is analyzed in two independent stages. Any changes made to the model parameters affecting the first stage of analysis changes the simulation data used in the second stage of analysis. Therefore, second stage parameters were much easier to adjust and re-analyze than the first stage.

The first stage consists of creating the subject profiles and determining the associated recognition and recall thresholds, as described above. Once the thresholds are set for a particular subject profile, the first stage of analysis is complete. The second stage of analysis involves running test simulations. Using the thresholds from the first stage, the simulated recognition and recall test results are calculated and compared to the experimental data.

This means that the subject profiles parameters (attention signal values and white noise), and other model parameters (e.g. number of neurons) which would affect the simulation output (i.e. threshold values) cannot be altered in the second stage of analysis without repeating the first stage of analysis as well. This created time constraints on the number of parameters that could be investigated, as the first stage of analysis requires many hours of simulation for each subject profile.

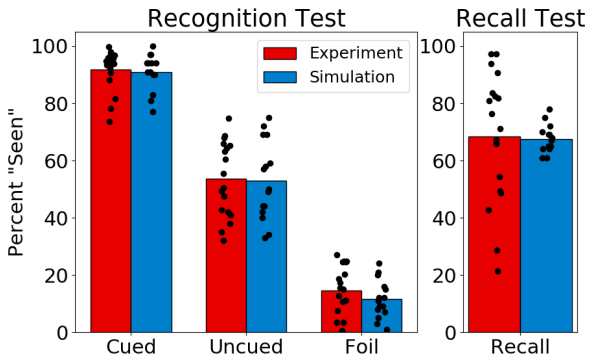


Figure 5: Comparison of experimental and simulated test results (John H. Wittig et al., 2018). In each case, only sessions that met both specified criteria were used (71/90 valid experimental sessions, 132/150 valid simulated sessions). Each data point represents the average recognition rate for an individual subject over 10 test sessions.

Table 2: Statistical Comparison of Test Data.

| Category | Cued | Uncued | Foil | Recalled |
|---|---|---|---|---|
| Experimental Variance | 50.7 | 174.3 | 79.7 | 534.3 |
| Simulation Variance | 39.4 | 190.9 | 44.4 | 23.3 |

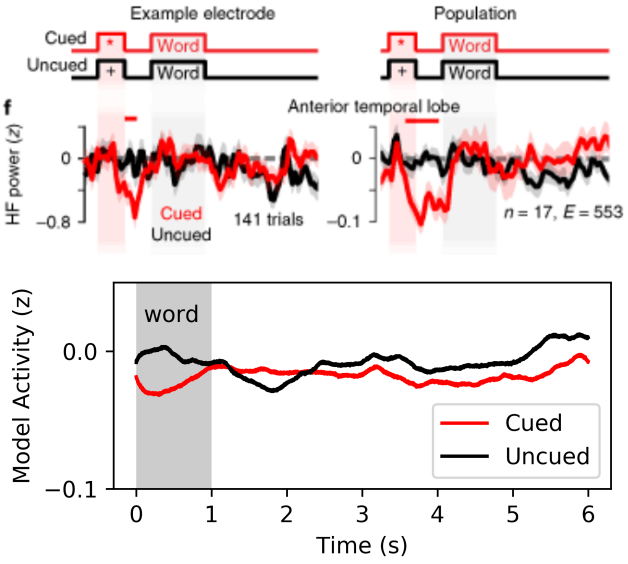


Figure 6: (TOP) Experimental neuron activity represented by high-frequency power electrode feedback, from (John H. Wittig et al., 2018). (Left) Single electrode. (Right) Population average across all electrodes. A red bar at the top of each plot indicates where there is a statistically significant difference between the cued and uncued responses. (BOTTOM) Average neuron activity from the model, for cued and uncued conditions. Note that the current model does not include the points in time between the cue and word presentation.

## Task Performance

The model's performance is compared to experiment in the rates of recognition and recall for each subject. Since the recognition and recall thresholds were generated using the experimental test data, it is expected that the simulated tests would produce similar average rates of recognition and recall. Therefore, we will focus instead on the variance of correct response rates for each of the two tests. These variances can be visually compared in Figure 5, and the calculated statistics are summarized in Table 2.

There is a large differences in variance between the experimental recognition and recall tests; however, the simulated results do not share this difference. Additionally, experimental and simulated sessions failed the performance criteria at approximately similar rates (Exp. = 79%, Sim. = 89%).

Figure 6 (Top) shows the activity of neurons measured by (John H. Wittig et al., 2018) during the experiment. It can be seen that the presence of a cue corresponds to a drop in activity immediately before presentation of the word. The red bar above the plot indicates where this effect is statistically significant. Figure 6 (Bottom) is the analogous plot showing neuron activity of our model. During the word presentation and the period after the word presentation, our model shows no difference in neural activity for cued versus uncued words. This is consistent with the empirical data. However, our model does not include the extra processing needed to *remember the cue*, and thus we do not expect to see a corresponding drop in activity in our model.

## Model Exploration

As a test to check for extreme over-fitting in recreating the experimental correct response rates, we also ran the participant profiles to list lengths both longer and shorter than what they were tuned for. Longer lists should produce lower rates of correct responses for tests involving task list words, while shorter list lengths should produce higher rates of correct responses in the same tests. There should be no significant effect on the foil word rates as this represents random chance. The results of these modifications can be seen in Figure 7.

## Discussion

In Figure 5 we see that the model behaves similarly to human subjects in the recognition test. We were able to choose a single cosine similarity threshold that produced results with significant differences in recognition rates between the cued, uncued and foil word categories, meeting the performance criteria. Table 2 illustrates that the variance in test performance is similar to experiment for the recognition test, but not for the recall test.

One possible explanation for the model's ability to match human behaviour well in the recognition test, but not the re-

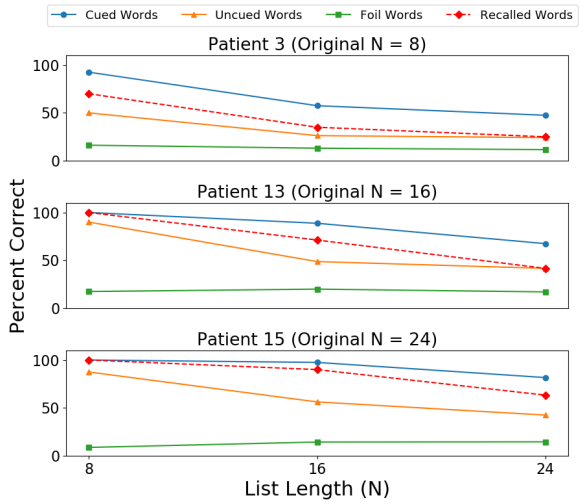


Figure 7: Effects of altering list length on task performance for three patients. The subject profile of patient 3 is calibrated for list lengths of N = 8, patient 13 for list lengths of N = 16, and patient 15 for list lengths of N = 24.

call test, is the way these tests were designed in the simulation. The simulated recognition test mirrors the experiment in that both involve presenting the subject with the word, then determining whether the subject recognizes that word. In the experiment, subjects are visually shown the word and must decide whether the word came from the original task list. In the simulation, the word vectors are compared to the final memory vector through cosine similarity. Conversely, in the experimental recall test subjects recall words from memory without a prompt, whereas in the simulated recall test cosine similarity is used again. This may indicate that using cosine similarity is a reasonable approximation of how the human brain performs the recognition test, but that humans use a different mechanism in performing the recall test.

The cosine similarity time plot (Figure 4) shows that the memory neurons of the model store information in an intuitive way. Words that are introduced to the memory space first are represented more strongly than those introduced later, regardless of whether or not they are cued. Therefore, the first word in a list will be imprinted most strongly on memory, since it is the only information stored in memory at that time. This effect is seen in Figure 4, where the first word is uncued, yet has the greatest similarity of all words when first introduced. After the fourth word of the list, the difference between high and low attention signals is more evident, as there are enough words affecting memory at this point that recency bias becomes less influential. This pattern reflects the way we would expect humans to remember words in a list.

Figure 7 illustrates that the model responds as expected when changing the difficulty of the task by altering the task list length for a subject profile. For cued and uncued words in the recognition test and cued words in the recall test, the simulation performance reduced as the task list length increased. This is the type of behaviour that we would expect from human subjects as the difficulty of the task, represented by task list length, increased. Performance for the foil words remained relatively flat, which is also as expected. This indicates that the fitting of the model parameters reasonably generalizes to other conditions, and produces predictions about individual performance on varying list lengths.

The simulation activity plot, Figure 6 (Bottom), does not show a difference between cued and uncued words. This is likely due to the design of the model. In the model, there is no analog for the cue, prior to the word being presented. The memory input simply represents a visual stimulus, multiplied by an attention value. This was done to keep the model simple, with the goal of comparing simulation and experimental results. In a further iteration of this model, a proper cue mechanism could be added examine the effects on model spiking activity.

Overall, we have presented a model showing a possible neural implementation of attention-driven memory. By taking the simple approach of representing words as vectors fed into a memory, and by scaling those vectors based on the amount of attention paid to them, we achieve a reasonable approximation of human behaviour in the recognition test, but the actual empirical data on the recall test has a much higher variance. Furthermore, we do not currently include the part of the task involving remembering the cue. Future work will address these concerns by including both the cue memory and the decision-making process to extract information out of the memory.

## References

Bekolay, T., Bergstra, J., Hunsberger, E., DeWolf, T., Stewart, T. C., Rasmussen, D., ... Eliasmith, C. (2014). Nengo: A python tool for building large-scale functional brain models. *Frontiers in Neuroinformatics*, *7*(48).

Eliasmith, C., & Anderson, C. H. (2003). *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. Cambridge, MA: MIT Press.

Eliasmith, C., Stewart, T. C., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., & Rasmussen, D. (2012). A large-scale model of the functioning brain. *Science*, *338*, 1202-1205.

Hurzook, A., Trujillo, O., & Eliasmith, C. (2013). Visual motion processing and perceptual decision making. In *35th annual conference of the cognitive science society* (pp. 2590–2595).

John H. Wittig, J., Jang, A. I., Cocjin, J. B., Inati, S. K., & Zaghloul, K. A. (2018). Attention improves memory by suppressing spiking-neuron activity in the human anterior temporal lobe. *Nature Neuroscience*, *21*, 808–810.

Sharma, S., Komer, B. J., Stewart, T. C., & Eliasmith, C. (2016). A neural model of context dependent decision making in the prefrontal cortex. In *38th annual meeting of the cognitive science society* (pp. 1122–1127). Austin, TX: Cognitive Science Society.