bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Spike-frequency adaptation provides a long short-term memory to networks of spiking neurons

Short title: Short-term memory using spike-frequency adaptation

Darjan Salaj[1,†], Anand Subramoney[1,†], Ceca Kraišniković[1,†], Guillaume Bellec[1,2], Robert Legenstein[1], Wolfgang Maass[1,*]

May 7, 2020

[1] *Institute of Theoretical Computer Science, Graz University of Technology, Inffeldgasse 16b, Graz, Austria*
[2] *Laboratory of Computational Neuroscience, Ecole Polytechnique Fédérale de Lausanne (EPFL), Bâtiment AAB, offices 135-141, CH-1015 Lausanne*
† *These authors contributed equally to this work.*
* *To whom correspondence should be addressed; E-mail: maass@igi.tugraz.at.*

Abstract

Brains are able to integrate memory from the recent past into their current computations, seemingly without effort. This ability is critical for cognitive tasks such as speech understanding or working with sequences of symbols according to dynamically changing rules. But it has remained unknown how networks of spiking neurons in the brain can achieve that. We show that the presence of neurons with spike frequency adaptation makes a significant difference: Their inclusion in a network moves its performance for such computing tasks from a very low level close to the level of human performance. While artificial neural networks with special long short-term memory (LSTM) units had already reached such high performance levels, they lack biological plausibility. We find that neurons with spike-frequency adaptation, which occur especially frequently in higher cortical areas of the human brain, provide to brains a functional equivalent to LSTM units.

Introduction

Our brains are able to constantly process new information in the light of recent experiences and dynamically changing rules, seemingly without any effort. But we do not know how networks of spiking neurons (SNNs) in the brain accomplish that. The performance of both spike-based and rate-based models for recurrent neural networks in the brain have stayed on a rather low performance level for such tasks, far below the performance level of the human brain and artificial neural network models. Artificial neural network models that perform well on such tasks use, instead of neuron models, a special type of unit called Long Short-Term Memory (LSTM) unit. LSTM units store information in registers — like a digital computer — where it remains without perturbance by network activity for an indefinite amount of time, until is is actively updated or recalled. Hence these LSTM units are not biologically plausible, and it has remained an open problem how neural networks in the brain achieve so high performance on cognitively demanding tasks that require integration of information from the recent past into current computational processing. We propose that the brain achieves this — at least for some tasks — without separating computation and short-term memory in different network modules: Rather it intertwines computing and memory with the help of inherent slow dynamic processes in neurons and synapses.

Arguably the most prominent internal dynamics of neurons on the time scale of seconds — which is particularly relevant for many cognitive tasks — is spike-frequency adaptation (SFA). It

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

is expressed by a substantial fraction of neocortical neurons (Allen Institute, 2018). SFA reduces the excitability of a neuron in response to its firing, see Fig. 1. Neurons with SFA have often been included in SNN models that aim at modelling the dynamics of brain networks (Gutkin and Zeldenrust, 2014), but not in computational studies. We show that neurons with SFA do in fact significantly enhance the computational power of SNNs. This is somewhat surprising, because on first sight their history dependence, which even varies strongly from neuron to neuron, tends to obstruct — rather than enhance — network computations. We propose that this may hold for hand-constructed circuits, whereas evolutionary and learning processes are able to exploit advantages of such diverse forms of SFA. We demonstrate this in SNN models for a series of demanding benchmark tasks for network computations that all require integration of information over time: Recalling features of fleeting sensory inputs, speech recognition, time series classification, and operations on sequences of symbols.

We also compare the performance of SNNs with SFA to the performance of SNNs that have a different type of slow hidden dynamics, although on a smaller time scale — short-term plasticity (STP) of synapses. But the contribution of synaptic short-term plasticity — especially synaptic facilitation — to computational performance turns out to be lower. Interestingly, the most common form of STP in synapses between pyramidal cells, synaptic depression, tends to provide better support for such computations than synaptic facilitation. References to the related literature can be found in the Discussion.

Altogether our results suggest that neurons with SFA provide to SNNs a similar performance boost for computations that require a long short-term memory as LSTM units do for artificial neural networks. Hence we refer to SNNs that contain neurons with SFA as Long short-term memory SNNs (LSNNs). Since the term short-term memory is more common in the literature on LSTM networks, but the term working memory is more common in the neuroscience literature, and both appear to refer to the same phenomena, we treat these two notions as synonyms and let their use depend on the context.

SNNs are currently of high interest not only for modelling neural networks of the brain, but also as a computing paradigm for drastically more energy-efficient computer hardware. Hence it is of interest to see that the performance of LSTM networks, and thereby many recent achievements in Artificial Intelligence, can be ported to spike-based computing hardware.

Results

Experimental data and a simple model for SFA

The SFA of a neuron is usually measured in terms of the gradual increase of interspike intervals in its spike response to a constant input drive. An example for such measurement is the Adaptation Index (AI) that is employed by the Allen Institute (Allen Institute, 2018), see Fig. 1A for samples of neurons with different AI, Fig. 1B for the distribution of AI values, and the Methods for the definition of the AI. These data suggest that the human neocortex has a larger fraction of neurons with SFA than the mouse neocortex, and that within the human brain the frontal lobe has a larger fraction than the temporal gyrus. The analysis of experimental data in (Pozzorini et al., 2013, 2015) lead to the conclusion that SFA takes place on multiple time scales, with a history dependence that lasts up to 20 s in neocortical neurons. Various models for adapting neurons have been proposed in (Gerstner et al., 2014; Teeter et al., 2018). We employ a very simple model for SFA, the generalized leaky integrate-and-fire model $GLIF_2$ from (Teeter et al., 2018; Allen Institute, 2017), which we will refer to as the ALIF (adaptive LIF) model. A practical advantage of this simple model is that it can be very efficiently simulated and is amenable to gradient descent training methods. It assumes that the firing threshold $A(t)$ contains a variable component $a(t)$ that increases by a fixed amount after each of its spikes $z(t)$ (Fig. 1C), and then decays exponentially back to 0. This variable threshold models the inactivation of voltage-dependent sodium channels in a qualitative manner. We write $z_j(t)$ for the spike output of neuron j, that switches from 0 to 1 at time t when the neuron fires at time t, and otherwise has value 0. With this notation one can

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

define the ALIF model by the equations:

$$\begin{aligned} A_j(t) &= v_{\text{th}} + \beta a_j(t), \\ a_j(t+\delta t) &= \rho_j a_j(t) + (1-\rho_j) z_j(t) \delta t, \end{aligned} \tag{1}$$

where v_{th} is the constant baseline of the firing threshold $A_j(t)$, and $\beta > 0$ scales the amplitude of the activity-dependent component. The parameter $\rho_j = \exp\left(\frac{-\delta t}{\tau_{a,j}}\right)$ controls the speed by which $a_j(t)$ decays back to 0, where $\tau_{a,j}$ is the adaptation time constant and δt is the duration of a discrete time step (which we chose to be 1 ms). An LSNN is a network of spiking neurons that contains some ALIF neurons (see Methods for details on neuron and synapse models). It typically suffices to use ALIF neurons with some spread of time constants $\tau_{a,j}$ around the required duration of working memory for solving a task (see Table S1 in Supplement for details on how the choice of adaptation time constant impacts performance).

Methods for training recurrent SNNs

We focused on the best performing training method for recurrent SNNs that is currently known: Backpropagation through time ($BPTT$) with the pseudo-derivative for spiking neurons from (Bellec et al., 2018b). While $BPTT$ is not assumed to be biologically plausible as an online learning method, results from training with $BPTT$ inform us about computational capabilities of different types of SNNs. They also inform us about performance levels that could in principle be attained through evolution. In order to test whether complex cognitive computations, such as the 12AX task , can in principle also be learnt by brain networks, we also trained the same LSNN with a biologically plausible learning method: *e-prop*. For LSNNs, *e-prop* tends to achieve in general an almost as good computational performance level as $BPTT$ (Bellec et al., 2019).

SFA provides a functionally powerful working memory for spike-based computing

Our brains are able to recall an image, even after having seen many other images in between. We wondered whether LSNNs would be able to model such fundamental working memory task. Note that remembering an image requires retaining substantially more than a single bit, even if it is encoded in a highly compressed form in a higher cortical area. In contrast, most models for working memory have focused on retaining just a single bit, and this memory content occurred during training and testing. We formulated our more demanding computational task as the STORE-RECALL task illustrated in Fig. 2. The network received a sequence of frames, each consisting of a vector of 20 binary features — arranged in a 4×5 grid (top of Fig. 2) — which were presented for 200 ms. Each frame can be seen as corresponding to the compressed representation of an image in a higher visual area such as IT. In addition, the network received occasional STORE and RECALL commands, marked in yellow and green in Fig. 2. The STORE command corresponds to directing attention to a particular frame of the input stream. The computational task was to reproduce, during a RECALL command, the feature vector that had been presented during the preceding STORE command. The delay between the STORE and RECALL commands was randomly chosen from the interval between 200 and 1600 ms.

We trained an LSNN that consisted of 500 ALIF neurons, whose firing thresholds had time constants of 800 ms, to solve this task. Sigmoidal readout neurons, one for each of the 20 binary input features, were trained to reproduce the feature value that had been present during STORE. Binary feature values were extracted by rounding the activity of readout neurons at the half-way (100 ms) mark of each 200 ms time window. A sample segment of a test trial is shown in Fig. 2, with the activity of input neurons at the top and the activation of readout neurons at the bottom. In order to probe the generalization capability of the LSNN we made sure that none of the patterns shown during testing had occurred during training, and in fact had a Hamming distance of at least 5 bits to all training patterns. Note that previous models for working memory only aimed at storing a single bit, and the model could only be tested for the same content for which it was trained. Here we require that the working memory can be used for content other than what was used during training. The resulting recall performance of the LSNN was 99.09%, i.e., 99.09% of

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

the stored feature vectors were perfectly reproduced during recall. This demonstrates that LSNNs have inherent high-dimensional working memory capabilities. SFA was essential for this, because the recall performance of a recurrent network of LIF neurons without SFA, trained in exactly the same way, stayed at chance level (see Methods). A closer inspection of the time course of firing thresholds of a sample subset of neurons in the LSNN provides insight into how LSNNs are able to solve this task: A pattern-specific subset S of neurons is highly activated during STORE, which raises their firing thresholds (shown as blue curves in Fig. 2). Many neurons are activated again during RECALL, but the firing activity of neurons in the subset S remains lower this time, thereby providing a negative imprint of their activation pattern during STORE. Readout neurons can easily be trained to decode these negative imprints, and to reproduce the originally stored pattern.

Interestingly, the firing activity of the network was rather low during the delay between STORE and RECALL. Furthermore we found that a Support Vector Machine (SVM) was not able to decode the stored feature vector from the firing activity of the LSNN neurons during the delay (the decoding accuracy during the delay was 4.38%, as opposed to 100% decoding accuracy during RECALL; see Methods). Hence the type of working memory that an LSNN exhibits corresponds to the activity-silent form of working memory in the human brain that had been examined in the experiments of (Wolff et al., 2017). It had also been shown there that the representation of working memory content changes drastically between memory encoding and subsequent network reactivation during the delay by an "impulse stimulus": A classifier trained on the network activity during encoding was not able to classify the memory content during a network reactivation, and vice versa. The same holds for our LSNN model (see Methods), since the reactivation of the network during RECALL provides a negative, rather than a positive imprint, of the high-dimensional memory content.

We also examined how the time constants of the thresholds of ALIF neurons should be chosen to achieve good performance for a task that requires a particular time span of working memory. We studied this for a variant of the STORE-RECALL task where the time-varying input vector of features was just 1D instead of 20D, but where the expected delays between STORE and RECALL varied between 0.2 to 16 s for different versions of the task. It turned out that good performance did not require a tight coupling between the required length of working memory and the adaptation time constants of ALIF neurons in the LSNN, see Table S1. In particular, good working memory performance was also achieved when the required time span for working memory was substantially larger than these time-constants, suggesting that the network had learned to automatically refresh or stagger the implicit memory in firing thresholds of different neurons. We also verified that good performance for many memory retention time spans could be achieved by a single network with a mixture of time constants of firing thresholds drawn from a uniform or power-law distribution. This suggests that brains can solve working memory tasks for many different retention spans using SFA neurons with a generic spread of time constants.

Finally, we wondered whether the adaptive firing threshold of ALIF neurons affects the auto-correlation function of their firing activity — termed intrinsic timescale in (Wasmuht et al., 2018). We tested this for an LSNN consisting of 200 LIF and 200 ALIF neurons that was trained to solve a 1D version of the STORE-RECALL task. It turned out that during the delay between STORE and RECALL these intrinsic time constants were in the same range as those measured in monkey cortex, see Fig. 1C in (Wasmuht et al., 2018). Furthermore LIF and ALIF neurons exhibited very similar distributions of these time constants (see Fig. S1), suggesting that these intrinsic time constants are determined largely by their network inputs, and less by the neuron type.

Working memory performance of variants of SNNs with other slow processes in neurons or synapses

There exist numerous other slow processes on the time scale of seconds in neurons and synapses, that can potentially also enhance network computations on this time scale. We examined the performance of three other candidates besides SFA on a simple version of the STORE-RECALL task:

- LIF neurons whose excitability gets increased through their firing: ELIF neurons

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

- Depressing short-term plasticity of synapses (STP-D)
- Facilitating short-term plasticity of synapses (STP-F).

The ELIF neuron is a dual version of an ALIF neuron whose excitability is increased through preceding firing, rather than decreased (see Methods). ELIF neurons appear to be particularly suitable for creating a working memory through persistent firing. Facilitating short-term plasticity of synapses also supports that, and was conjectured by (Mongillo et al., 2008) to produce a working memory. We also evaluated the performance of depressing short-term plasticity, because this is the standard dynamics of synapses between pyramidal cells (Markram et al., 2015). The dynamics of the salient hidden variables in these three models is illustrated in Fig. S2. The performance of corresponding variants of the SNN is shown in Fig. 3A for a 1D variant of the STORE-RECALL task from Fig. 2, with a delay between STORE and RECALL commands varying between 200 ms and 3600 ms. It turns out that SNNs with ALIF neurons, i.e., LSNNs, learn to solve this task much faster than the other variants of the SNN model, and also reach the highest performance level. Furthermore only the networks with STP-D or ELIF neurons eventually approach reasonably good — although lower — performance levels. We were surprised to see that facilitating short-term plasticity of synapses (STP-F) did not provide the working memory capability needed to solve this task, although we used here a really long time constant for facilitation with a mean of 2000 ms — much larger than the mean of 507 ms that had been found experimentally in (Wang et al., 2006) for synaptic connections between pyramidal cells in the PFC. Similar results hold for the time series classification task sMNIST, see Fig. 3C and the subsection on sMNIST below.

We also found that replacing ALIF by ELIF neurons reduced the working memory capability of the network for both tasks, see Fig. 3A and C. One possible reason is that information that is stored in the firing threshold of a neuron is better protected in the case of an ALIF neuron, since an increased firing threshold suppresses subsequent accidental firing, and hence accidental modifications of the memory that is stored in the firing threshold. In contrast, for an ELIF neuron the information that is stored in the firing threshold is quite vulnerable, since a decreased firing threshold invites accidental firing.

Performance of LSNNs for speech recognition and other benchmark tasks that require substantial integration of information over time

Google Speech Commands dataset. We trained LSNNs and networks of LIF neurons on the keyword spotting task with Google Speech Commands Dataset (Warden, 2018) (v0.02). The dataset consists of 105,000 audio recordings of people saying thirty different words. Fully connected networks were trained to classify audio recordings, that are clipped to one second length, into one of 12 classes (10 keywords, as well as two special classes for silence and unknown words; the remaining 20 words had to be classified as "unknown"). Comparison of maximum performance of trained spiking networks against state-of-the-art artificial recurrent networks is shown in Table 1. Averaging over 5 runs, the LSNN reached $90.88 \pm 0.22\%$, and the LIF network reached $88.79 \pm 0.16\%$ accuracy. Thus an SNN without ALIF neurons can already solve this task quite well, but the LSNN halves the performance gap to the published state-of-the-art in machine learning. The only other report on a solution of this task with spiking networks is (Cramer et al., 2019). There the authors encode the audio features to spike trains using cochlea model and train a network of LIF neurons using surrogate gradients with $BPTT$ and achieve $50.9 \pm 1.1\%$ accuracy.

Delayed-memory XOR task. We also tested the performance of LSNNs on a previously considered benchmark task for SNNs, where two items in the working memory have to be combined non-linearly: The Delayed-memory XOR task (Huh and Sejnowski, 2018). The network is required to compute the exclusive-or operation on the history of input pulses when prompted by a go-cue signal, see Fig. 3B.

The network receives on one input channel two types of pulses (up or down), and a go-cue on another channel. If the network received two input pulses since the last go-cue signal, it should generate the output "1" during the next go-cue if the input pulses were different or "0" if the input pulses were the same. Otherwise, if the network only received one input pulse since the last go-cue signal, it should generate a null output (no output pulse). Variable time delays are introduced

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

between the input and go-cue pulses. Time scale of the task was 600 ms which limited the delay between input pulses to 200 ms.

This task was solved in (Huh and Sejnowski, 2018), without providing a performance statistics, by using a type of neuron that has not been documented in biology — a non-leaky quadratic integrate and fire neuron. We are not aware of previous solutions by networks of LIF neurons. To compare and investigate the impact of SFA on the performance of delayed-memory XOR task, we trained networks of ALIF and LIF neurons of the same size as in (Huh and Sejnowski, 2018) — 80 neurons. Across 10 runs, networks of ALIF neurons solved the task with $95.19 \pm 0.014\%$ accuracy, whereas the networks of LIF neurons converged at lower $61.30 \pm 0.029\%$ accuracy.

Sequential MNIST (sMNIST).

Finally, we compared the performance of LSNNs and other variants of SNNs with that of LSTM networks on a more demanding benchmark task for time series classification: The classification of pixel-wise sequentially presented handwritten digits (sMNIST dataset), see Fig. 3C,D and Fig. S3. This task requires integration of information over a longer time span than for recognizing speech commands. It also requires very good generalization capability, since the pixel sequences for different handwriting styles of the same digit may vary widely. LSNNs achieved here about the same performance level as LSTM networks, whereas networks that contain only LIF and not ALIF neurons performed poorly, see Fig. 3D. Besides a fully connected LSNN, we also tested the performance of a variant of the model, called SC-LSNN, that integrates additional constraints of SNNs in the brain: It is sparsely connected (12% of possible connections are present) and consists of 75% excitatory and 25% inhibitory neurons that adhere to Dale's law. By adapting the sparse connections with the rewiring method in (Bellec et al., 2018a) during *BPTT* training, the SC-LSNN was enabled to perform even better than the fully-connected LSNN. The resulting architecture of the SC-LSNN is shown in Fig. S3C. Its activity of excitatory and inhibitory neurons, as well as the time courses of adaptive thresholds for (excitatory) ALIF neurons of the SC-LSNN are shown in Fig. S3B.

Fig. 3C shows that, apart from LSNNs, SNNs with experimentally reported parameters for short-term synaptic plasticity (STP-D) also achieve very high performance. Furthermore, SNNs with STP-D perform substantially better for this task than networks with data-based synaptic facilitation (STP-F), similar as for STORE-RECALL (see Fig. 3A).

SFA supports demanding cognitive computations with dynamically changing rules

The 12AX task — which can be viewed as a simplified version of the Wisconsin Card Sorting task — tests the capability of subjects to apply dynamically changing rules for pattern recognition and to ignore currently irrelevant inputs (O'Reilly and Frank, 2006; MacDonald III, 2008). It also probes — at least in the more demanding version that we consider — the capability to maintain and update a hierarchical working memory. The task consists of a sequence of trials where the subject is first shown a context cue to indicate which one of two possible sequences is the "correct" symbol sequence in the current trial. These sequences consist of two relevant symbols, interspersed with distractor symbols, including those belonging to the "wrong" sequence in the current context. At every step, the subject has to press one of two buttons depending on whether the correct sequence has been completed or not. The context of the trial switches randomly after a few presentation of symbols.

To model this, we gave as network inputs sequences of 90 symbols from the set {1, 2, A, B, C, X, Y, Z}, with repetitions as described in Methods. See the top of Fig. 4 for an example. After each symbol, the network should output "R" if this symbol terminates a context dependent target sequence and "L" otherwise. Specifically, given a context where the most recently received digit was a "1", the subject should output "R" only after presentation of a symbol X that terminates a subsequence A...X. This occurs, for example, for the 7th symbol in the trial shown in Fig. 4. In case that the most recent input digit was a "2", the subject should respond "R" only after the symbol Y in a subsequent subsequence B...Y (see the 20th symbol in Fig. 4). The letters C and Z are irrelevant and serve as distractors. This task requires a hierarchical working memory, because the most recently occurring digit determines whether subsequent occurrences of "A" or "B" should

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

be placed into working memory. Note also that neither the content of the higher-level working memory — the digit — nor the content of the lower level working memory — the letter A or B — are simply recalled. Instead, they both affect processing rules, where the higher-level processing rule affects what is placed into the lower level working memory. A simpler version of this task, where X and Y were relevant only if they directly followed A or B respectively, and where fewer irrelevant letters occurred in the input, was solved in (O'Reilly and Frank, 2006) through artificial neural network models that were endowed with special working memory modules. Note that for this simpler version no lower order working memory is needed, because one just has to wait for an immediate transition from A to X in the input sequence, or for an immediate transition from B to Y. But neither the simpler nor the more complex version of the 12AX-task has previously been solved by a network of spiking neurons.

We show in Fig. 4 that a generic LSNN can solve this quite demanding version of the 12AX task. The LSNN received spike trains from the input population of spiking neurons, producing Poisson spike trains. Possible input symbols {1, 2, A, B, C, X, Y, Z} were encoded using one-hot coding; each input symbol was signaled through a high firing rate of a separate subset of 5 input neurons for 500 ms. The LSNN consisted of 200 recurrently connected spiking neurons (100 ALIF and 100 LIF neurons), with all-to-all connections between them. The output consisted of two readouts, one for L, one for the R response. During each 500 ms time window the input to these readouts was the average activity of neurons in the LSNN during that time window. The final output symbol was based on which of the two readouts had the maximum value. After training with *BPTT* the LSNN produced an output string with all correct symbols in 97.79% of episodes, where 90 symbols had to be processed during each episode. But also after training with *e-prop*, a biologically realistic learning method (Bellec et al., 2019), the LSNN produced fully correct output sequences in 92.89% of the episodes. In contrast, a recurrent SNN with the same architecture but no neurons with SFA could achieve only 0.39% fully correct output strings after training with *BPTT* (not shown). Note that it was not necessary to create a special network architecture for the two levels of working memory that our more complex version of the 12AX task requires: A near perfectly performing network emerged from training a generic LSNN. This shows that neurons with SFA enable generic recurrent networks of spiking neurons to solve demanding cognitive tasks involving dynamically changing rules and two levels of working memory.

SFA supports brain-like operations on sequences

A generic difficulty for neural networks is learning to carry out operations on sequences of symbols in such a way that they generalize to new sequences, a fundamental capability of the human brain (Marcus, 2003). Actually, not only humans, but also non-human primates are able to carry out operations on sequences of items, and numerous neural recordings — starting with (Barone and Joseph, 1989) up to recent results such as (Carpenter et al., 2018; Liu et al., 2019) — provide information about the neural codes for sequences that accompany such operations in the brain. One fundamental question is how serial order of items is encoded in working memory. Behind this is the even more basic question of how transient structural information — the serial position of an item — is combined in the brain with content information about the identity of the item (Lashley, 1951). Obviously, this question also lies at the heart of open questions about the interplay between neural codes for syntax and semantics that enable language understanding in the human brain. The experimental data both of (Barone and Joseph, 1989) and (Liu et al., 2019) suggest that the brain uses a factorial code where position and identity of an item in a sequence are encoded separately by some neurons, thereby facilitating flexible generalization of learned experience to new sequences. But so far we had been lacking spiking neural network models that were able to carry out such tasks, and whose emergent neural codes could then be compared with neural recordings from the brain. We show here that LSNNs can be trained to carry out complex operations on sequences, are able to generalize such capabilities to new sequences, and produce spiking activity and neural codes that offer interesting links to recorded data.

One basic operation on sequences of symbols is remembering and reproducing a given sequence (Liu et al., 2019), a task which non-human primates can also learn, and for which neural codes for sequences have been investigated (Barone and Joseph, 1989; Liu et al., 2019). A more complex operation that can also be carried out by the brain is the reversal of a sequence (Marcus, 2003;

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Liu et al., 2019). We show that an LSNN learns to carry out both of these operations, and is able to apply them to new sequences of symbols that did not occur during training.

We trained an LSNN consisting of 128 LIF and 192 ALIF neurons to carry out these two operations on sequences of 5 symbols from a repertoire of 31 symbols, which we labeled by the letters a, b, c, ..., x, y, z, A, B, C, D, E from the English alphabet. Four additional symbols were used: "*" denoted the end of the input sequence (EOS), "?" a prompt for an output symbol, and one symbol each for the DUPLICATE and REVERSE commands (see Fig. 5A). Each of the altogether 35 input symbols were given to the network in the form of higher firing activity of a dedicated population of 5 input neurons outside of the LSNN ("one hot encoding"). It was not necessary to assign particular values to adaptation time constants of firing thresholds of neurons with SFA; we simply chose them uniformly randomly to be between 1 ms and 6000 ms, mimicking the diversity of SFA effects found in the neocortex (Allen Institute, 2018) in a qualitative manner. The network output was produced by linear readouts (one per potential output symbol, each with a low pass filter with a time constant of 250 ms) that received spikes from neurons in the LSNN, see the row "Output" in Fig. 5A). The final output symbol was selected using the readout which had the maximum value at the end of each 500 ms time window (a softmax instead of the hard argmax was used during training), mimicking winner-take-all computations in neural circuits of the brain (Chettih and Harvey, 2019) in a qualitative manner.

After training, an LSNN was able to apply duplication and reversal also to new sequences, achieving a success rate of 0.9588 (average over 5 random seeds) for unseen sequences. The "success rate" was defined as the fraction of test episodes/trials where the full output sequence was generated correctly. Sample episodes of the trained LSNN are shown in Fig. 5A. For comparison, we also trained a LIF network in exactly the same way with the same number of neurons. It achieved a performance of 0.0 (zero).

Emergent coding properties of neurons in the LSNN are analyzed in Fig. 5B-F. Neurons are sorted in Fig. 5B,C according to the time of their peak activity (averaged over 1000 episodes), like in (Harvey et al., 2012). A number of network neurons (about one-third) participate in sequential firing activity independent of the type of task and the symbols involved (Fig. 5B). Instead, these neurons have learned to abstract the overall timing of the tasks. This kind of activity is reminiscent of the neural activity relative to the start of a trial that was recorded in rodents after they had learned to solve tasks that had a similar duration (Tsao et al., 2018).

The time of peak activity of other neurons depended on the task and the concrete content, see Fig. 5C. Interestingly enough, these neurons change their activation order already during the loading of the input sequence in dependence of the task (duplication or reversal). Using 3-way ANOVA, we were able to categorize each neuron as selective to a specific condition or a non-linear combination of conditions based on the effect size ω^2. Each neuron could belong to more than one category if the effect size was above the threshold of 0.14 (as suggested by (Field, 2013)). Similar to recordings from the brain (Carpenter et al., 2018), a diversity of neural codes emerged that encode one or several of the variables symbol identity, serial position in the sequence, and type of task. In other words, a large fraction of neurons are mixed-selective, i.e. selective to non-linear combinations of all three variables. Peri-condition time histogram (PCTH) plots of two sample neurons are shown in Fig. 5E,F: One neuron is selective to symbol "g" but at different positions depending on task context. The other neuron is selective to symbol "C" occurring at position 5 in the input, independent of task context. Thus one sees that a realization of this task by an SNN, which was previously not available, provides rich opportunities for a comparison of emergent spike codes in the model and neuronal recordings from the brain.

Discussion

An important open problem in computational neuroscience is to understand how brains carry out computations that involve not just current cues, but information from the recent past. In fact, brains are able to store not just single bits for subsequent computational use over a time scale of many seconds, but previously experienced images, movie scenes, and dialogues that require a fairly large storage capacity. This problem is usually formulated as a question about the implementation of working memory — or short-term memory — in the brain. But this formulation is somewhat

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

biased against the possibility that computing and short-term memory are so intertwined in neural networks of the brain that it becomes really difficult to separate mechanisms and network modules that hold short-term memory from those that constitute the computational machinery of the network.

There already exists fairly wide agreement that different forms of working or short-term memory can be distinguished in the brain, see e.g. (Olivers et al., 2011; Kamiński and Rutishauser, 2019; Masse et al., 2019). Recent experimental data show clearly that, for a highly trained task, discrete attractors of the network dynamics, implemented by persistent firing, hold an intended movement direction in the anterior lateral motor cortex (Inagaki et al., 2019). But the question remains whether the brain uses the same mechanism — especially without extensive training — for holding quickly changing high-dimensional memory content, such as a movie scene, previously read text, or a sequence of images. Neural codes for storing information from a sequences of two images — after extensive training — had been examined in (Warden and Miller, 2007). A complex interaction was found between the memory traces of two sequentially presented images, thereby speaking against an assumption that each is held by a separate discrete attractor in working memory.

Several publications argue that the brain uses a variety of mechanisms for working memory, each with its specific advantages and disadvantages, whose engagement depends on the specific task (Olivers et al., 2011; Trübutschek et al., 2017; Kamiński and Rutishauser, 2019; Barbosa et al., 2019; Hu et al., 2020). In particular, (Wolff et al., 2017; Trübutschek et al., 2017; Kamiński and Rutishauser, 2019; Masse et al., 2019; Barbosa et al., 2019) point to an activity-silent form of working memory that is used by the brain for maintaining working memory while it is not in the focus of attention. A model for such activity-silent memory had been proposed already in (Mongillo et al., 2008), based on facilitating short-term plasticity of synapses. This mechanism requires a facilitating short-term plasticity of synapses between excitatory neurons (pyramidal cells), which had previously been discovered in the medial prefrontal cortex of ferret (Wang et al., 2006). However the model of (Mongillo et al., 2008) used a time constant of 1500 ms for the time constant F of facilitation, whereas this parameter for the facilitation-dominant synapse type E1 of (Wang et al., 2006) has a reported average value of 507 ms with a standard deviation of 37 ms. An experimentally testable prediction of this form of activity-silent working memory is that an unspecific network reactivation between storage and recall would make the content of working memory decodable from the resulting network activity. However, the experimental data of (Wolff et al., 2017) do not support this prediction.

We examined in this paper whether the arguably most prominent dynamic feature of neurons on the time-scale of seconds, SFA, supports computations that require a working memory. Experimental data show that SFA does in fact produce history-dependence of neural firing on a time scale of several seconds up to 20 seconds (Pozzorini et al., 2013, 2015). We found that this prominent feature of a fairly large fraction of neurons in the neocortex provides an inherent working memory capability to neural networks. Our results suggest that this working memory capability is functionally quite powerful, and enables networks of spiking neurons to solve a variety of cognitively demanding tasks that were previously beyond the reach of SNN models. In particular, SFA enables flexible operations on sequences of symbols (Fig. 4, 5). This allows us, for the first time, to study emergent neural codes for symbols and their position in a sequence in a model network of spiking neurons, and to compare them with recordings from neurons in the neocortex for corresponding tasks (Fig. 5). When we compared the contribution of SFA with the contribution of the other two most prominent slow processes in neurons or synapses, synaptic facilitation and synaptic depression, we found that the contribution of SFA is substantially more powerful for a basic working memory task (Fig. 3A). A comparison for a demanding time series classification task with a lower demand on the retention time span (Fig. 3C) suggests that synaptic depression works for such tasks about equally well, but not synaptic facilitation. The good performance of synaptic depression for tasks that require shorter retention time of working memory is consistent with the modelling results of (Masse et al., 2019) and (Hu et al., 2020). However, as already pointed out in (Masse et al., 2019), synaptic depression tends to work best for tasks that require rather short working memory maintenance. Our results are also consistent with the finding of (Masse et al., 2019) that persistent activity is more prominent if the working memory content has to be manipulated, rather than just maintained. Compare the higher firing activity in Fig. S3B for the sMNIST task that requires continuous manipulation of working memory content with the low firing activity in Fig. 2,

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

where the working memory content just has to be maintained. One sees this difference also in the sequence manipulation task of Fig. 5. There the working memory just has to be maintained during the first half of a trial, yielding an average firing rate of 16.6 Hz over all neurons. But this average firing rate increased to 26.7 Hz during the second halfs of the trials, where the stored information had to be manipulated (averages taken over 50,000 trials during testing).

On first sight one might think that working memory can only be held in neural activity through increased firing. Our results show that it can just as well be attained through decreased firing, which is the way how neurons with SFA provide evidence of preceding strong activation. Whereas this mechanism may be intuitively less plausible, it looks equally viable from the perspective of downstream networks in the brain. Whether preceding firing activity leaves a positive or negative imprint in subsequent firing appears to be of secondary relevance for readout neurons if the downstream integration of evidence involves a weighted sum, since weights can have positive or negative signs. One may argue that there are actually, from the systems-perspective, two benefits in maintaining working memory in the form of a negative imprint, i.e., through decreased excitability of neurons. One is that encoding working memory through non-firing consumes less energy. Another is that this form of working memory is less vulnerable to disturbances through intervening network activity, since a decreased excitability protects a neuron from accidental activation — and hence potential overwriting of its memory content. One first piece of experimental evidence for the negative imprinting hypothesis was provided by the previously mentioned result of (Wolff et al., 2017). It was examined there whether a classifier that had been trained to decode from the network activity the stored memory content during encoding would be able to decode the memory content also during a subsequent network reactivation through an unspecific impulse. The answer was negative, which is consistent with the negative imprinting hypothesis. We confirmed for the task of Fig. 2 that the same holds true for our model with SFA (see subsection "Decoding memory from the network activity" in Methods). It is actually well-known that negative imprinting is used by the brain for a particular type of long-term memory called recognition memory: Familiarity of an object is encoded through reduced firing of a large fraction of neurons in the perirhinal cortex and adjacent areas, see (Winters et al., 2008) for a review.

A major structural difference between standard models for neural networks in the brain and artificial neural networks (ANNs) that are used in artificial intelligence and deep learning for solving computational tasks that involve memory from the recent past lies in the type of neurons (units) that are used. Well-performing ANNs usually employ LSTM units or similar units that allow to store a bit or analog variable in a memory register — like in a digital computer — where it is protected from perturbation by ongoing network activity. Our results show — rather surprisingly — that such drastic protection of working memory content is not needed: We showed that almost the same performance can be achieved by LSNNs, i.e., SNNs that contain neurons with SFA. This holds in spite of the fact that memory content that is stored in an adaptive firing threshold of a neuron with SFA is not fully protected from the disturbance through network activity. But it is at least somewhat shielded, because a neuron that holds memory in the form of an increased firing threshold has an inherent tendency not to respond to smaller membrane depolarizations.

Our results show that biologically rather realistic models for spiking neural networks in the brain that also contain neurons with SFA reach for demanding cognitive tasks for the first time the performance level of humans, which could previously only be reached with ANNs that employ biologically unrealistic LSTM units. This paves the way for reaching a key-goal of brain modelling — to combine detailed experimentally data from neurophysiology on the level of neurons and synapses with brain-like functionality of the network.

Recurrent networks of spiking neurons are also of interest from the perspective of novel computing technology. Spike-based computing hardware has the potential to provide substantially more energy-efficient implementations of artificial intelligence and deep learning results than standard digital hardware. But its performance has so far been significantly inferior to that of non-spiking neural networks. Our results show that this performance gap is becoming quite small for the case of recurrent neural networks if one integrates neurons with SFA into the spike-based network.

Altogether, we have shown that a well-known feature of a substantial fraction of neurons in the neocortex — SFA — provides an important new facet to our understanding of computations in SNNs: It enables SNNs to integrate working memory from the recent past seamlessly into ongoing network computations.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Materials and Methods

Table of Contents:

- Adaptation Index
- Network models
- Training methods
- Tasks

Adaptation index

The adaptation index (AI) is a quantitative measure of firing rate adaptation that has been recorded for a wide variety of cells in the Allen institute database (Allen Institute, 2018). It measures the rate at which firing of a spiking neuron speeds up or slows down when the neuron is fed with a step current of 1 second. Given the induced spike times, it is defined as:

$$\frac{1}{N-1}\sum_{n=1}^{N-1}\frac{\mathrm{ISI}_{n+1}-\mathrm{ISI}_n}{\mathrm{ISI}_{n+1}+\mathrm{ISI}_n},$$

where ISI_n is n-th inter spike interval (ISI) and N is the number of ISIs induced during the stimulus duration. Hence regular doubling of the ISI produces for example $\mathrm{AI} = 0.33$.

Network models

Leaky integrate and fire (LIF) neurons. A LIF neuron j spikes as soon at its membrane potential $V_j(t)$ is above its threshold v_{th}. At each spike time t, the membrane potential $V_j(t)$ is reset by subtracting the threshold value v_{th} and the neuron enters a strict refractory period for 2 to 5 ms (depending on the experiment) where it cannot spike again. Between spikes the membrane voltage $V_j(t)$ is following the dynamic:

$$\tau_m \dot{V}_j(t) \quad = \quad -V_j(t) + R_m I_j(t).$$

Our simulations were performed in discrete time with a time step $\delta t = 1$ ms. In discrete time, the input and output spike trains are modeled as binary sequences $x_i(t), z_j(t) \in \{0, \frac{1}{\delta t}\}$ respectively. Neuron j emits a spike at time t if it is currently not in a refractory period, and its membrane potential $V_j(t)$ is above its threshold. During the refractory period following a spike, $z_j(t)$ is fixed to 0. The neural dynamics in discrete time reads as follows:

$$V_j(t+\delta t) \quad = \quad \alpha V_j(t) + (1-\alpha) R_m I_j(t) - v_{\mathrm{th}} z_j(t)\delta t, \tag{2}$$

where $\alpha = \exp(-\frac{\delta t}{\tau_m})$, with τ_m being the membrane constant of the neuron j. The spike of neuron j is defined by $z_j(t) = H\left(\frac{V_j(t)-v_{\mathrm{th}}}{v_{\mathrm{th}}}\right)\frac{1}{\delta t}$, with $H(x) = 0$ if $x < 0$ and 1 otherwise. The term $-v_{\mathrm{th}} z_j(t)\delta t$ implements the reset of the membrane voltage after each spike.

In all simulations the R_m was set to $1\,\mathrm{G}\Omega$. The input current $I_j(t)$ is defined as the weighted sum of spikes from external inputs and other neurons in the network:

$$I_j(t) \quad = \quad \sum_i W^{in}_{ji} x_i(t - d^{in}_{ji}) + \sum_i W^{rec}_{ji} z_i(t - d^{rec}_{ji}), \tag{3}$$

where W^{in}_{ji} and W^{rec}_{ji} denote respectively the input and the recurrent synaptic weights and d^{in}_{ji} and d^{rec}_{ji} the corresponding synaptic delays.

Adaptive leaky integrate and fire (ALIF) neurons. An ALIF neuron extends the LIF neuron with an SFA mechanism. The SFA is realized by replacing the fixed threshold v_{th} with the adaptive threshold $A_j(t)$ which follows the dynamic described in equation (1). The spiking output of ALIF neuron j is then defined by $z_j(t) = H(\frac{V_j(t)-A_j(t)}{A_j(t)})\frac{1}{\delta t}$.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Adaptation time constants of ALIF neurons were chosen to match the task requirements while still conforming to the experimental data from rodents (Allen Institute, 2018; Pozzorini et al., 2013, 2015; Mensi et al., 2012). For an analysis of the impact of the adaptation time constants on the performance see Table S1 in Supplement.

LIF neurons whose excitability gets increased through their firing: ELIF neurons. There exists experimental evidence that some neurons fire for the same stimulus more for a repetition of the same sensory stimulus. We refer to such neurons as ELIF neurons, since they are becoming more excitable. Such repetition enhancement was discussed for example in (Tartaglia et al., 2015). But to the best of our knowledge, it has remained open whether repetition enhancement is a network effect, resulting for example from a transient depression of inhibitory synapses onto the cell that is caused by postsynaptic firing (Kullmann et al., 2012), or a result of an intrinsic firing property of some neurons. We used a simple model for ELIF neurons that is dual to the ALIF neuron model: The threshold is lowered through each spike of the neuron, and then decays exponentially back to its resting value. This can be achieved by using a negative value for β in equation (1).

Models for Short-Term Plasticity (STP) of synapses. We modelled the STP dynamic according to the classical model of STP in (Mongillo et al., 2008). The STP dynamics in discrete time, derived from the equations in (Mongillo et al., 2008), are as follows:

$$u'_{ji}(t+\delta t) = \exp\left(\frac{-\delta t}{F}\right) u'_{ji}(t) + U_{ji}(1-u_{ji}(t))z_i(t)\delta t, \tag{4}$$

$$u_{ji}(t+\delta t) = U_{ji} + u'_{ji}(t), \tag{5}$$

$$r'_{ji}(t+\delta t) = \exp\left(\frac{-\delta t}{D}\right) r'_{ji}(t) + u_{ji}(t)(1-r'_{ji}(t))z_i(t)\delta t, \tag{6}$$

$$r_{ji}(t+\delta t) = 1 - r'_{ji}(t), \tag{7}$$

$$W^{STP}_{ji}(t+\delta t) = W^{rec}_{ji} u_{ji}(t) r_{ji}(t), \tag{8}$$

where $z_i(t)$ is the spike train of the pre-synaptic neuron and W^{rec}_{ji} scales the synaptic efficacy of synapses from neuron i to neuron j. Networks with STP were constructed from LIF neurons with the weight W^{rec}_{ji} in equation (3) replaced by the time dependent weight $W^{STP}_{ji}(t)$.

STP time constants of facilitation-dominant and depression-dominant network models were based on values of experimental recordings in (Wang et al., 2006) of PFC-E1 and PFC-E2 synapse types respectively. Recordings in (Wang et al., 2006) were performed in medial prefrontal cortex of young adult ferrets. For the STORE-RECALL task, both facilitation and depression time constants were equally scaled up until the larger time constant matched the requirement of the task (see section on "Comparing networks with different slow processes" below). In the sMNIST task for the depression-dominant network model (STP-D) we used values based on PFC-E2: $F = 20$ ms, $D = 700$ ms and $U = 0.2$, and for facilitation-dominant network model (STP-F) we used values based on PFC-E1: $F = 500$ ms, $D = 200$ ms and $U = 0.2$.

Weight initialization. Initial input and recurrent weights were drawn from a Gaussian distribution $W_{ji} \sim \frac{w_0}{\sqrt{n_{in}}}\mathcal{N}(0,1)$, where n_{in} is the number of afferent neurons and $\mathcal{N}(0,1)$ is the zero-mean unit-variance Gaussian distribution and $w_0 = \frac{1\,\mathrm{Volt}}{R_m}\delta t$ is a normalization constant (Bellec et al., 2018b).

Training methods

BPTT. In artificial recurrent neural networks such as LSTMs, gradients can be computed with backpropagation through time ($BPTT$). In spiking neural networks, complications arise from the non-differentiability of the output of spiking neurons. In our discrete time simulation, this is formalized by the discontinuous step function H arising in the definition of the spike variable $z_j(t)$. All other operations can be differentiated exactly with $BPTT$. For feedforward artificial neural networks using step functions, a solution was to use a pseudo derivative $H'(x) := \max\{0, 1-|x|\}$,

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

but we observed that this convention is unstable with recurrently connected neurons. We found that dampening this pseudo-derivative with a factor $\gamma < 1$ (typically $\gamma = 0.3$) solves that issue. Hence we use the pseudo-derivative:

$$\frac{dz_j(t)}{dv_j(t)} := \gamma \max\{0, 1 - |v_j(t)|\}, \tag{9}$$

where $v_j(t)$ denotes the normalized membrane potential $v_j(t) = \frac{V_j(t) - A_j(t)}{A_j(t)}$. Importantly, gradients can propagate in adaptive neurons through many time steps in the dynamic threshold without being affected by the dampening.

e-prop. In the 12AX task the networks were trained using biologically plausible learning method *random e-prop* (Bellec et al., 2019) in addition to *BPTT*.

Tasks

The STORE-RECALL task of Fig. 2 The input to the network consisted of STORE, RECALL, and 20 bits which were represented by sub-populations of spiking input neurons. STORE and RECALL commands were represented by 4 neurons each. The 20 bits were represented by population coding where each bit was assigned 4 input neurons (2 for value zero, and 2 for value one). When a sub-population is active, it would exhibit a Poisson firing with frequency of 400 Hz. To measure the generalization capability of a trained network, we first generate a test set dictionary of 20 unique feature vectors (random bit strings of length 20) that have at least a Hamming distance of 5 bits among each other. For every training batch a new dictionary of 40 random bit strings (of length 20) would be generated where each string has a Hamming distance of at least 5 bits from any of the bit string in the test set dictionary. This way we ensure that, during training, a network never encounters any bit string similar to one from the test set. Each input sequence consisted of 10 steps (200 ms each) where a different population encoded bit string is shown during every step. Only during the RECALL period, the 20 bit input populations are silent. At every step, the STORE or the RECALL populations were activated interchangeably with probability 0.2 which resulted in distribution of delays between the STORE-RECALL pairs in the range [200, 1600] ms.

The training and the test performance were computed as average over 256 and 512 random input sequences respectively. Networks were trained for 4000 iterations and stopped if the error on the training batch was below 1%. We used the Adam optimizer with default parameters and initial learning rate of 0.01 which is decayed every 200 iterations by a factor of 0.8. We also used learning rate ramping, which, for the first 200 iterations, monotonically increased the learning rate from 0.00001 to 0.01. To avoid unrealistically high firing rates, the loss function contained a regularization term (scaled with coefficient 0.001) that minimizes the squared difference of the average firing rate of individual neurons from a target firing rate of 10 Hz. To improve convergence, we also included an entropy component to the loss (scaled with coefficient 0.3) which was computed as the mean of the entropies of the sigmoid neurons outputs.

We trained an ALIF network and a LIF network, both consisting of 500 recurrently connected neurons. The membrane time constant was $\tau_m = 20$ ms. For adaptation parameters we used $\beta_{ALIF} = 4$ mV and $\tau_a = 800$ ms with baseline threshold voltage 10 mV. Synaptic delay was 1 ms. The input to the sigmoidal readout neurons were the neuron traces that were calculated by passing all the network spikes through a low-pass filter with a time constant of 20 ms.

We ran 5 training runs with different random seeds for both ALIF and LIF network models. All runs of the ALIF network converged after ~ 3600 iterations to a training error below 1%. At that point we measured the accuracy on 512 test sequences generated using the previously unseen test bit strings which resulted in test accuracy of 99.09% with standard deviation of 0.17%. The LIF network was not able to solve the task in any of the runs (all runs resulted in 0% training and test accuracy with zero standard deviation). On the level of individual feature recall accuracy, the best out of 5 training runs of the LIF network was able to achieve 49% accuracy which is the chance level since individual features are binary bits. In contrast, all ALIF network runs had individual feature level accuracy of above 99.99%.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Decoding memory from the network activity. We trained a Support Vector Machine (SVM) to classify the stored memory content from the network spiking activity in the step before the RECALL (200 ms before the start of RECALL command). We performed a cross-validated grid-search to find the best hyperparameters for the SVM which included kernel type [linear, polynomial, RBF] and penalty parameter C of the error term [0.1, 1, 10, 100, 1000]. We trained SVMs on test batches of the 5 different training runs (see above). SVMs trained on the period preceding the RECALL command of a test batch achieved an average of 4.38% accuracy with standard deviation of 1.29%. In contrast SVMs trained on a period during the RECALL command achieved an accuracy of 100%. This demonstrates that the memory stored in the network is not decodable from the network firing activity before the RECALL input command.

Additionally, analogous to the experiments of (Wolff et al., 2017), we trained SVMs on network activity during the encoding (STORE) period and evaluated them on the network activity during reactivation (RECALL), and vice versa. In both scenarios, the classifiers were not able to classify the memory content of the evaluation period (0.0% accuracy).

Comparing networks with different slow processes on a simplified version of the STORE-RECALL task. For the comprehensive comparison of networks endowed with different slow processes in neuron and synapse dynamics we used a single dimensional version of the STORE-RECALL task where only a single feature needs to be stored and recalled from memory. The input to the network consisted of 40 input neurons: 10 for STORE, 10 for RECALL, and 20 for population coding of a binary feature. Each sequence consisted of 20 steps (200 ms each) where the STORE or the RECALL populations were activated with probability 0.09 interchangeably which resulted in delays between the STORE-RECALL pairs to be in the range [200, 3600] ms.

The training batch and the test performance were computed as average over 128 and 2048 random input sequences respectively. All networks were trained for 400 iterations. We used the Adam optimizer with default parameters and initial learning rate of 0.01 which was decayed every 100 iterations by a factor of 0.3. The same firing rate regularization term was added to the loss as in the original STORE-RECALL setup (see above).

All networks consisted of 60 recurrently connected neurons. The membrane time constant was $\tau_m = 20\,\text{ms}$. For ALIF and ELIF networks, we used $\beta_{ALIF} = 1\,\text{mV}$ and $\beta_{ELIF} = -0.5\,\text{mV}$ with $\tau_a = 2000\,\text{ms}$. Synapse parameters of STP-D network were $F = 51 \pm 15$ ms, $D = 2000 \pm 51$ ms and $U = 0.25$, and of STP-F network $F = 2000 \pm 146$ ms, $D = 765 \pm 71$ ms and $U = 0.28$. The baseline threshold voltage was 10 mV for all models except ELIF for which it was 20 mV. Synaptic delay was 1 ms across all network models.

Google Speech Commands task. Features were extracted from the raw audio using the Mel Frequency Cepstral Coefficient (MFCC) method with 30 ms window size, 1 ms stride and 40 output features. The network models were trained to classify the input features to one of the 10 keywords (yes, no, up, down, left, right, on, off, stop, go) or to two special classes for silence or unknown word (where the remainder of 20 recorded keywords are grouped). The training, validation and test set were assigned 80, 10, and 10 percent of data respectively while making sure that audio clips from the same person stay in the same set.

All networks were trained for 18,000 iterations using the Adam optimizer with batch size 100. The output spikes of the networks were averaged over time, and the linear readout layer was applied to those values. During the first 15,000 we used a learning rate of 0.001 and for the last 3000 we used a learning rate of 0.0001. The loss function contained a regularization term (scaled with coefficient 0.001) that minimizes the squared difference of average firing rate between individual neurons and a target firing rate of 10 Hz.

Both ALIF and LIF networks consisted of 2048 fully connected neurons in a single recurrent layer. The neurons had a membrane time constant of $\tau_m = 20\,\text{ms}$, the adaptation time constant of ALIF neurons was $\tau_a = 100\,\text{ms}$, adaptation strength was $\beta = 2\,\text{mV}$. The baseline threshold was $v_{\text{th}} = 10\,\text{mV}$, and the refractory period was 2 ms. Synaptic delay was 1 ms.

Delayed-memory XOR task. The pulses on the two input channels were generated with 30

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

ms duration and the shape of a normal probability density function normalized in the range $[0, 1]$. The pulses were added or subtracted from the baseline zero input current at appropriate delays. The go-cue was always a positive current pulse. The 6 possible configurations of the input pulses ($+$, $-$, $++$, $--$, $+-$, $-+$) were sampled with equal probability during training and testing.

Networks were trained for 2000 iterations using the Adam optimizer with batch size 256. The initial learning rate was 0.01 and every 200 iterations the learning rate was decayed by a factor of 0.8. The loss function contained a regularization term (scaled with coefficient 50) that minimizes the squared difference of the average firing rate of individual neurons from a target firing rate of 10 Hz. This regularization resulted in networks with mean firing rate of 10 Hz where firing rates of individual neurons were spread in the range [1, 16] Hz.

Both ALIF and LIF networks consisted of 80 fully connected neurons in a single recurrent layer. The neurons had a membrane time constant of $\tau_m = 20\,\text{ms}$, a baseline threshold $v_{\text{th}} = 10\,\text{mV}$, and a refractory period of 3 ms. ALIF neurons had an adaptation time constant of $\tau_a = 500\,\text{ms}$ and an adaptation strength of $\beta = 1\,\text{mV}$. Synaptic delay was 1 ms. For training the network to classify the input into one of the three classes, we used the cross-entropy loss between the labels and the softmax of three linear readout neurons. The input to the linear readout neurons were the neuron traces that were calculated by passing all the network spikes through a low-pass filter with a time constant of 20 ms.

The sequential MNIST (sMNIST) task. The input consisted of sequences of 784 pixel values created by unrolling the handwritten digits of the MNIST dataset, one pixel after the other in a scanline manner as indicated in Fig. S3A. For comparing different spiking network models, we used 1 ms presentation time for each pixel (Fig. 3C). LSTM networks also work well for tasks on larger time-scales. Hence for comparing LSNNs with LSTM networks, we used a version of the task with 2 ms presentation time per pixel, thereby doubling the length of sequences to be classified to 1568 ms (Fig. 3D). A trial of a trained LSNN (with an input sequence that encodes a handwritten digit "3" using population rate coding) is shown in Fig. S3B. The top row of Fig. S3B shows a version where the grey value of the currently presented pixel is encoded by population coding, through the firing probability of 80 input neurons. Somewhat better performance was achieved when each of the 80 input neurons was associated with a particular threshold for the grey value, and this input neuron fired whenever the grey value crossed its threshold in the transition from the previous to the current pixel (this input convention was used for the results of Fig. 3C,D). Grey values of pixels were presented to the LSTM network simply as analog numbers.

Networks were trained for 36,000 iterations using the Adam optimizer with batch size 256. The initial learning rate was 0.01 and every 2500 iterations the learning rate was decayed by a factor of 0.8. The loss function contained a regularization term (scaled with coefficient 0.1) that minimizes the squared difference of average firing rate between individual neurons and a target firing rate of 10 Hz.

The neurons had a membrane time constant of $\tau_m = 20\,\text{ms}$, a baseline threshold of $v_{\text{th}} = 10\,\text{mV}$, and a refractory period of 5 ms. The adaptation time constants of ALIF and ELIF neurons were $\tau_a = 700\,\text{ms}$ in Fig. 3C. ALIF neurons had $\tau_a = 1400\,\text{ms}$ in Fig. 3D. The adaptation strength of ALIF neurons was $\beta = 1.8\,\text{mV}$, and of ELIF neurons $\beta = -0.9\,\text{mV}$. Synaptic delay was 1 ms. The output of the LSNN is produced by the softmax of 10 linear output neurons that receive spikes from all neurons in the network, as shown in the bottom row of Fig. S3B. For training the network to classify to one of the ten classes we used cross-entropy loss computed between the labels and the softmax of output neurons. The input to the linear readout neurons were the neuron traces that were calculated by passing all the network spikes through a low-pass filter with a time constant of 20 ms.

The 12AX task. The input for each training and testing episode consisted of a sequence of 90 symbols from the set {1,2,A,B,C,X,Y,Z}. A single episode could contain multiple occurrences of digits 1 or 2 (up to 23), each time changing the target sequence (A...X or B...Y) after which the network was supposed to output R. Each digit could be followed by up to 26 letters before the next digit appeared. More precisely, the following regular expression describes the string that is produced: `[12][ABCXYZ]{1,10}((A[CZ]{0,6}X|B[CZ]{0,6}Y)|([ABC][XYZ])){1,2}`. Each choice in this regular expression is made randomly.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

The neurons had a membrane time constant of $\tau_m = 20\,\text{ms}$, a baseline threshold $v_{\text{th}} = 30\,\text{mV}$, a refractory period of 5 ms, and synaptic delays of 1 ms. ALIF neurons had an adaptation strength of $\beta = 1.7\,\text{mV}$, and adaptation time constants were chosen uniformly from $[1, 13500]$ ms.

A cross-entropy loss function was used along with a regularization term (scaled with coefficient 15) that minimizes the squared difference of average firing rate between individual neurons and a target firing rate of 10 Hz. The LSNN was trained for 10,000 iterations with a batch size of 20 episodes and a fixed learning rate of 0.001. An episode consisted of 90 steps, with between 4 to 23 tasks generated according to the task generation procedure described previously. We trained the network with $BPTT$ using 5 different seeds, which resulted in average test success rate 97.79% with standard deviation 0.42%. The network trained with *random e-prop* using 5 different seeds resulted in average test success rate 92.89% with standard deviation 0.75%.

Symbolic computation on strings of symbols. The input to the network consisted of 35 symbols - 31 symbols represented symbols from the English alphabet {a, b, c, d, ... x, y, z, A, B, C, D, E}, one symbol was for "end-of-string" (EOS) '*', one for cue for the output prompt '?', and two symbols to denote whether the task instruction was duplication or reversal. The task and the rest of the symbols were encoded using separate one-hot vectors of dimension 2 and 33 respectively. Inputs to the network were transformed into spike trains using a population of 5 spiking neurons for each input component for a total of 175 input neurons. This population fired at a "high" rate (200 Hz) to encode 1, and at a "low" rate (2 Hz) otherwise. The output consisted of 32 linear readouts: 31 for symbols from the English alphabet and one additional readout for the "end-of-string" symbol. The input to these linear readouts was the value of neuron traces at the end of each step of 500 ms during the output period, i.e, the second half of each episode. The neuron traces were calculated by passing all the network spikes through a low-pass filter with a time constant of 250 ms. The final output symbol was produced using the argmax over the value of all the readouts (a softmax instead of the hard argmax was used during training). The network was trained to minimize the cross entropy error between the softmax applied on the output layer and targets. The loss function contained a regularization term (scaled with coefficient 5) that minimizes the squared difference of average firing rate between individual neurons and a target firing rate of 20 Hz.

The training was performed for 50,000 iterations, with a batch size of 50 episodes. We used the Adam optimizer with default parameters and fixed learning rate of 0.001. Each symbol was presented to the network for a duration of 500 ms. The primary metric we used for measuring the performance of the network was success rate, which was defined as the percentage of episodes where the network produced the full correct output for a given string i.e. all the output symbols in the episode had to be correct. The network was tested on 50,000 previously unseen strings.

The network consisted of 128 LIF and 192 ALIF neurons. All the neurons had a membrane time constant of $\tau_m = 20\,\text{ms}$, a baseline threshold $v_{\text{th}} = 30\,\text{mV}$, a refractory period of 5 ms, and a synaptic delay of 1 ms. ALIF neurons in the network had an adaptation strength of $\beta = 1.7\,\text{mV}$, and an adaptation time constant randomly uniformly chosen from the range $[1, 6000]$ ms. All other parameters were the same as in the other experiments. We trained the network using 5 different seeds and tested it on previously unseen strings. Average test success rate was 95.88% with standard deviation 1.39%.

Analysis of spiking data. We used 3-way ANOVA to analyze if a neuron's firing rate is significantly affected by task, serial position in the sequence, symbol identity, or combination of these (similar to (Lindsay et al., 2017)). We refer to these factors as "conditions". The analysis was performed on the activity of the neurons of the trained LSNN during 50,000 test episodes. For the analysis, neurons whose average firing rate over all episodes was lower than 2Hz or greater than 60Hz were discarded from the analysis to remove large outliers. This left 279 out of the 320 neurons. From each episode, a serial position from the input period was chosen randomly, and hence each episode could be used only once, i.e., as one data point. This was to make sure that each entry in the 3-way ANOVA was completely independent of other entries, since the neuron activity within an episode is highly correlated. Each data point was labeled with the corresponding triple of (task type, serial position, symbol identity). To ensure that the dataset was balanced, the same number of data points per particular combination of conditions was used, discarding

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

all the excess data points, resulting in a total of 41,850 data points. To categorize a neuron as selective to one or more conditions, or combination of conditions, we observed p-values obtained from 3−way ANOVA and calculated the effect size ω^2 for each combination of conditions. If the p-value was smaller than 0.001 and ω^2 greater than 0.14 for a particular combination of conditions, the neuron was categorized as selective to that combination of conditions. The ω^2 threshold of 0.14 was suggested by (Field, 2013) to select large effect sizes. Each neuron can have large effect size for more than one combination of conditions. Thus the values shown in Fig. 5D sum to > 1. The neuron shown in Fig. 5E had the most prominent selectivity for the combination of Task $\times$ Position $\times$ Symbol, with $\omega^2 = 0.394$ and $p < 0.001$. The neuron shown in Fig. 5F was categorized as selective to a combination of Position $\times$ Symbol category, with $\omega^2 = 0.467$ and $p < 0.001$. While the 3-way ANOVA tells us if a neuron is selective to a particular combination of conditions, it does not give us the exact task/symbol/position that the neuron is selective to. To find the specific task/symbol/position that the neuron was selective to, Welch's t-test was performed, and a particular combination with maximum t-statistic and $p < 0.001$ was chosen to be shown in Fig. 5E,F.

References

Allen Institute. Allen Cell Types Database Technical white paper: GLIF models `http://help.brain-map.org/download/attachments/8323525/glifmodels.pdf`. Technical report, October 2017. v4.

Allen Institute. © 2018 Allen Institute for Brain Science. Allen Cell Types Database, cell feature search. Available from: `celltypes.brain-map.org/data`. 2018.

J. Barbosa, H. Stein, R. Martinez, A. Galan, K. Adam, S. Li, J. Valls-Solé, C. Constantinidis, and A. Compte. Interplay between persistent activity and activity-silent dynamics in prefrontal cortex during working memory. *bioRxiv*, page 763938, 2019.

P. Barone and J.-P. Joseph. Prefrontal cortex and spatial sequencing in macaque monkey. *Experimental brain research*, 78(3):447–464, 1989.

G. Bellec, D. Kappel, W. Maass, and R. Legenstein. Deep rewiring: Training very sparse deep networks. In *International Conference on Learning Representations*, Feb. 2018a. URL `https://openreview.net/forum?id=BJ_wN01C-`.

G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems*, pages 787–797, 2018b.

G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *bioRxiv*, page 738385, 2019.

A. F. Carpenter, G. Baud-Bovy, A. P. Georgopoulos, and G. Pellizzer. Encoding of serial order in working memory: neuronal activity in motor, premotor, and prefrontal cortex during a memory scanning task. *Journal of Neuroscience*, 38(21):4912–4933, 2018.

S. N. Chettih and C. D. Harvey. Single-neuron perturbations reveal feature-specific competition in V1. *Nature*, 567(7748):334–340, 2019.

B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke. The Heidelberg spiking datasets for the systematic evaluation of spiking neural networks. *arXiv preprint arXiv:1910.07407*, 2019.

A. Field. *Discovering statistics using IBM SPSS statistics*. Sage, 2013.

W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

B. Gutkin and F. Zeldenrust. Spike frequency adaptation. *Scholarpedia*, 9(2):30643, 2014. doi: 10.4249/scholarpedia.30643. revision #143322.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

C. D. Harvey, P. Coen, and D. W. Tank. Choice-specific sequences in parietal cortex during a virtual-navigation decision task. *Nature*, 484(7392):62–68, 2012.

B. Hu, M. E. Garrett, P. A. Groblewski, D. R. Ollerenshaw, J. Shang, K. Roll, S. Manavi, C. Koch, S. R. Olsen, and S. Mihalas. Adaptation supports short-term memory in a visual change detection task. *bioRxiv*, 2020.

D. Huh and T. J. Sejnowski. Gradient descent for spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 1433–1443, 2018.

H. K. Inagaki, L. Fontolan, S. Romani, and K. Svoboda. Discrete attractor dynamics underlies persistent activity in the frontal cortex. *Nature*, 566(7743):212–217, 2019.

J. Kamiński and U. Rutishauser. Between persistently active and activity-silent frameworks: novel vistas on the cellular basis of working memory. *Annals of the New York Academy of Sciences*, 2019.

D. M. Kullmann, A. W. Moreau, Y. Bakiri, and E. Nicholson. Plasticity of inhibition. *Neuron*, 75 (6):951–962, 2012.

A. Kusupati, M. Singh, K. Bhatia, A. Kumar, P. Jain, and M. Varma. Fastgrnn: A fast, accurate, stable and tiny kilobyte sized gated recurrent neural network. In *Advances in Neural Information Processing Systems*, pages 9017–9028, 2018.

K. S. Lashley. *The problem of serial order in behavior*, volume 21. Bobbs-Merrill Oxford, United Kingdom, 1951.

G. W. Lindsay, M. Rigotti, M. R. Warden, E. K. Miller, and S. Fusi. Hebbian learning in a random network captures selectivity properties of the prefrontal cortex. *Journal of Neuroscience*, 37(45): 11021–11036, 2017.

Y. Liu, R. J. Dolan, Z. Kurth-Nelson, and T. E. Behrens. Human replay spontaneously reorganizes experience. *Cell*, 178(3):640–652, 2019.

A. W. MacDonald III. Building a clinically relevant cognitive task: case study of the ax paradigm. *Schizophrenia bulletin*, 34(4):619–628, 2008.

G. F. Marcus. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. MIT Press, 2003.

H. Markram, E. Muller, S. Ramaswamy, M. W. Reimann, M. Abdellah, C. A. . Sanchez, and G. A. A. Kahou. Reconstruction and simulation of neocortical microcircuitry. *Cell*, 163(2): 456–492, 2015.

N. Y. Masse, G. R. Yang, H. F. Song, X.-J. Wang, and D. J. Freedman. Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nature Neuroscience*, page 1, 2019.

S. Mensi, R. Naud, C. Pozzorini, M. Avermann, C. C. Petersen, and W. Gerstner. Parameter extraction and classification of three cortical neuron types reveals two distinct adaptation mechanisms. *Journal of neurophysiology*, 107(6):1756–1775, 2012.

G. Mongillo, O. Barak, and M. Tsodyks. Synaptic theory of working memory. *Science*, 319(5869): 1543–1546, 2008.

C. N. Olivers, J. Peters, R. Houtkamp, and P. R. Roelfsema. Different states in visual working memory: When it guides attention and when it does not. *Trends in cognitive sciences*, 15(7): 327–334, 2011.

R. C. O'Reilly and M. J. Frank. Making working memory work: a computational model of learning in the prefrontal cortex and basal ganglia. *Neural computation*, 18(2):283–328, 2006.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

C. Pozzorini, R. Naud, S. Mensi, and W. Gerstner. Temporal whitening by power-law adaptation in neocortical neurons. *Nature neuroscience*, 16(7):942, 2013.

C. Pozzorini, S. Mensi, O. Hagens, R. Naud, C. Koch, and W. Gerstner. Automated high-throughput characterization of single neurons by means of simplified spiking models. *PLoS computational biology*, 11(6), 2015.

E. M. Tartaglia, G. Mongillo, and N. Brunel. On the relationship between persistent delay activity, repetition enhancement and priming. *Frontiers in psychology*, 5:1590, 2015.

C. Teeter, R. Iyer, V. Menon, N. Gouwens, D. Feng, J. Berg, A. Szafer, N. Cain, H. Zeng, M. Hawrylycz, et al. Generalized leaky integrate-and-fire models classify multiple neuron types. *Nature communications*, 9(1):1–15, 2018.

D. Trübutschek, S. Marti, A. Ojeda, J.-R. King, Y. Mi, M. Tsodyks, and S. Dehaene. A theory of working memory without consciousness or sustained activity. *Elife*, 6:e23871, 2017.

A. Tsao, J. Sugar, L. Lu, C. Wang, J. J. Knierim, M. B. Moser, and E. I. Moser. Integrating time from experience in the lateral entorhinal cortex. *Nature*, 561(7721):57–52, 2018.

Y. Wang, H. Markram, P. H. Goodman, T. K. Berger, J. Ma, and P. S. Goldman-Rakic. Heterogeneity in the pyramidal network of the medial prefrontal cortex. *Nature neuroscience*, 9(4):534, 2006.

M. R. Warden and E. K. Miller. The representation of multiple objects in prefrontal neuronal delay activity. *Cerebral Cortex*, 17(suppl_1):i41–i50, 2007.

P. Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

D. F. Wasmuht, E. Spaak, T. J. Buschman, E. K. Miller, and M. G. Stokes. Intrinsic neuronal dynamics predict distinct functional roles during working memory. *Nature communications*, 9 (1):3499, 2018.

B. D. Winters, L. M. Saksida, and T. J. Bussey. Object recognition memory: neurobiological mechanisms of encoding, consolidation and retrieval. *Neuroscience & Biobehavioral Reviews*, 32 (5):1055–1070, 2008.

M. J. Wolff, J. Jochim, E. G. Akyürek, and M. G. Stokes. Dynamic hidden states underlying working-memory-guided behavior. *Nature Neuroscience*, 20(6):864, 2017.

Acknowledgements

General: We would like to thank Pieter Roelfsema and Chris Summerfield for detailed comments on an earlier version of the manuscript.

Funding: This research was partially supported by the Human Brain Project (Grant Agreement number 785907) and the SYNCH project (Grant Agreement number 824162) of the European Union. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Quadro P6000 GPU used for this research. Computations were carried out on the Human Brain Project PCP Pilot Systems at the Juelich Supercomputing Centre, which received co-funding from the European Union (Grand Agreement number 604102) and on the Vienna Scientific Cluster (VSC).

Author contributions: The study was conceived by DS, AS, GB, WM. The experiments were designed by DS, AS, GB, RL, WM and were conducted by DS, AS, CK. The manuscript was written by DS, AS, CK, RL, GB, WM.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Competing interests: The authors declare no competing interests.

Data and materials availability: An implementation of the network model in Tensorflow/Python is available at `https://github.com/IGITUGraz/LSNN-official/`. The sMNIST dataset is available at `https://www.tensorflow.org/datasets/catalog/mnist`. The google speech commands dataset is available at `https://storage.cloud.google.com/download.tensorflow.org/data/speech_commands_v0.02.tar.gz`.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Figures and Tables

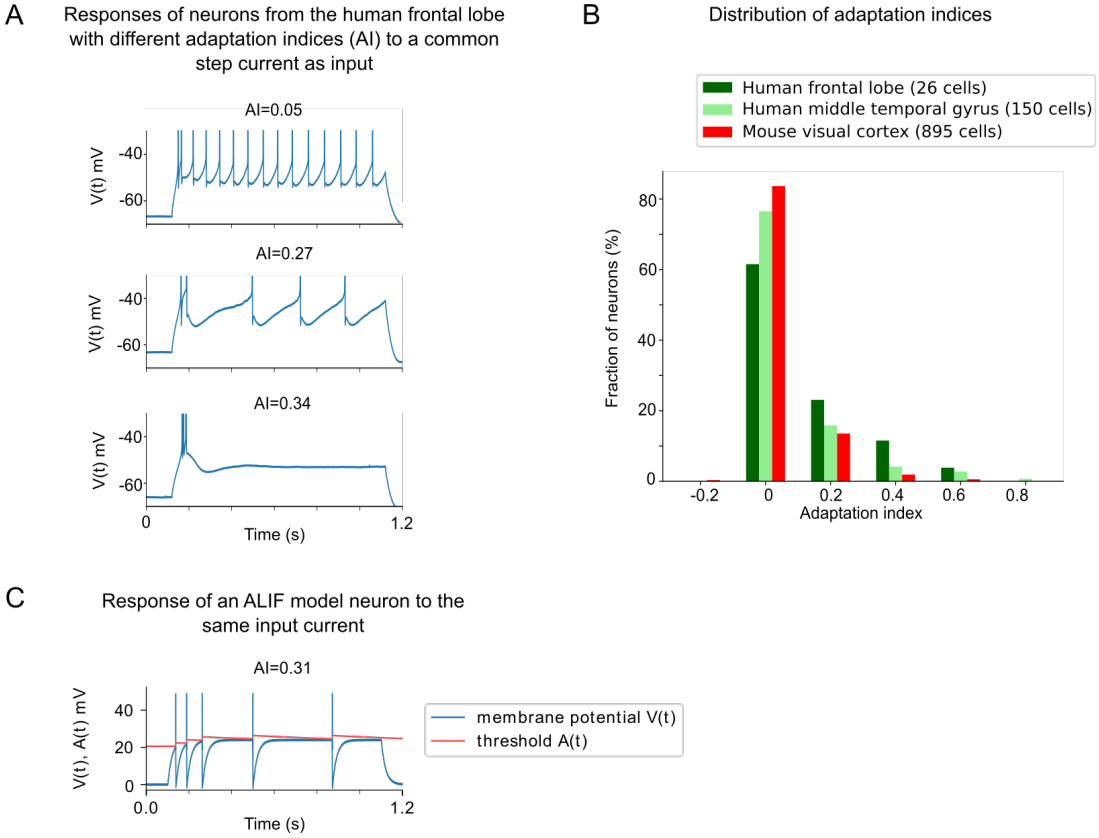

Figure 1: Experimental data on neurons with SFA, and a simple model for SFA. (**A**) The response to a 1-second long step current is displayed for three sample neurons in the neocortex. The adaptation index AI measures the rate of increase of interspike intervals. AI > 0 means that a neuron exhibits SFA. (**B**) Distribution of adaptation indices in neurons from human and rodent neocortex. Source of data for A and B: (Allen Institute, 2018). (**C**) Response of a simple model for a neuron with SFA — the adaptive LIF (ALIF) model — to the same input current as in A.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

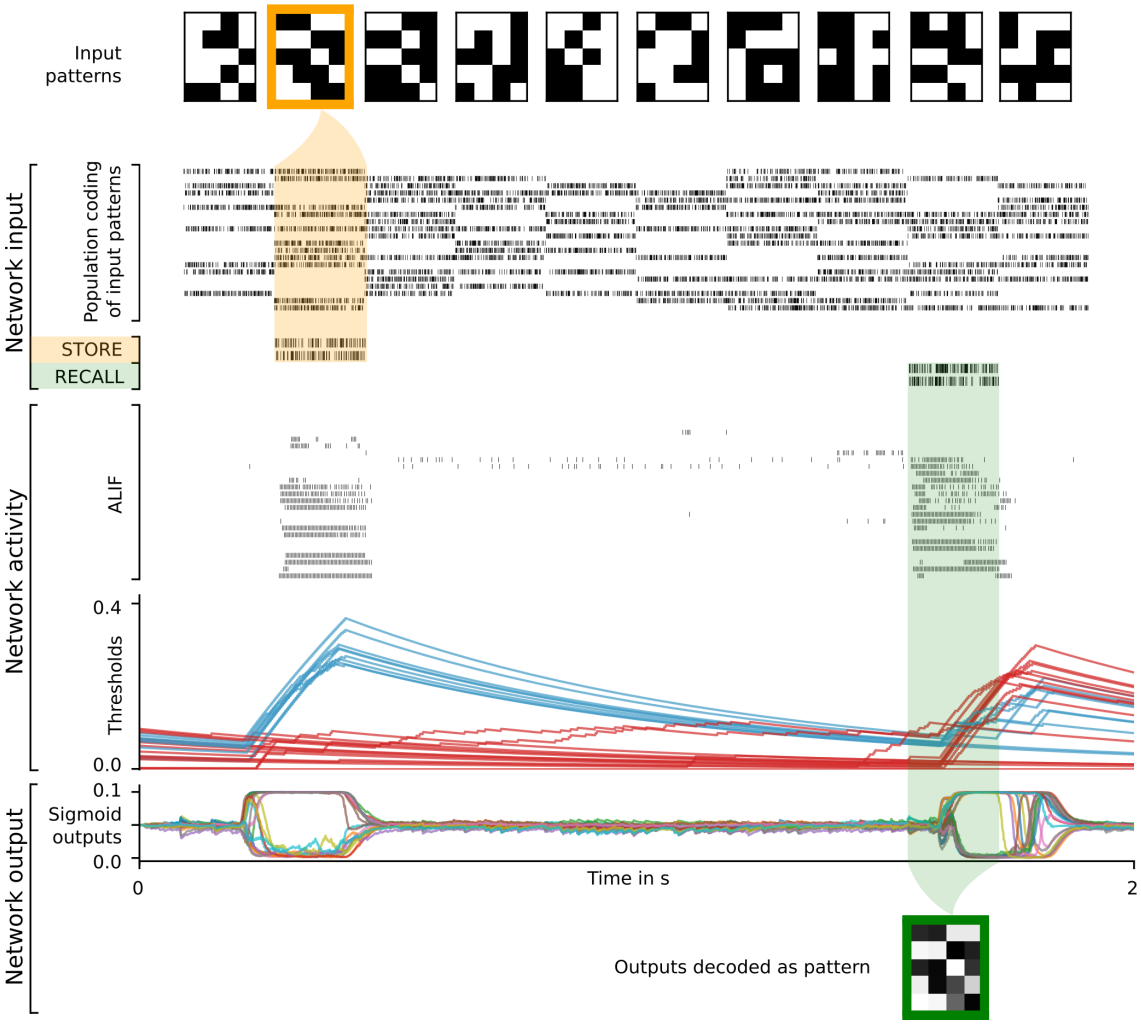

Figure 2: High-dimensional working memory capability of an LSNN. Rows top to bottom: Stream of randomly drawn 20 dimensional input patterns, represented by the firing activity of 20 populations of input neurons (subsampled), firing activity of two additional populations of input neurons for the STORE and RECALL commands, firing activity of 25 sample ALIF neurons in the LSNN (we first ordered all ALIF neurons with regard to the variance of their dynamic firing thresholds, and then picked every 20th), temporal evolution of the firing thresholds of these 25 neurons, traces of the activation of 20 sigmoidal readout neurons, and their average value during the 200 ms time window of the RECALL command represented by grey values. During the RECALL command (green shading) the network successfully reproduced the pattern that had been given as input during the preceding STORE command (yellow shading). Coloring of the threshold traces in blue or red was done after visual inspection to highlight the emergent two disjoint populations of ALIF neurons. The activity of one of them peaks during the STORE command, and provides a negative imprint of the stored pattern during RECALL through a reduced firing response. The other one peaks during RECALL.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

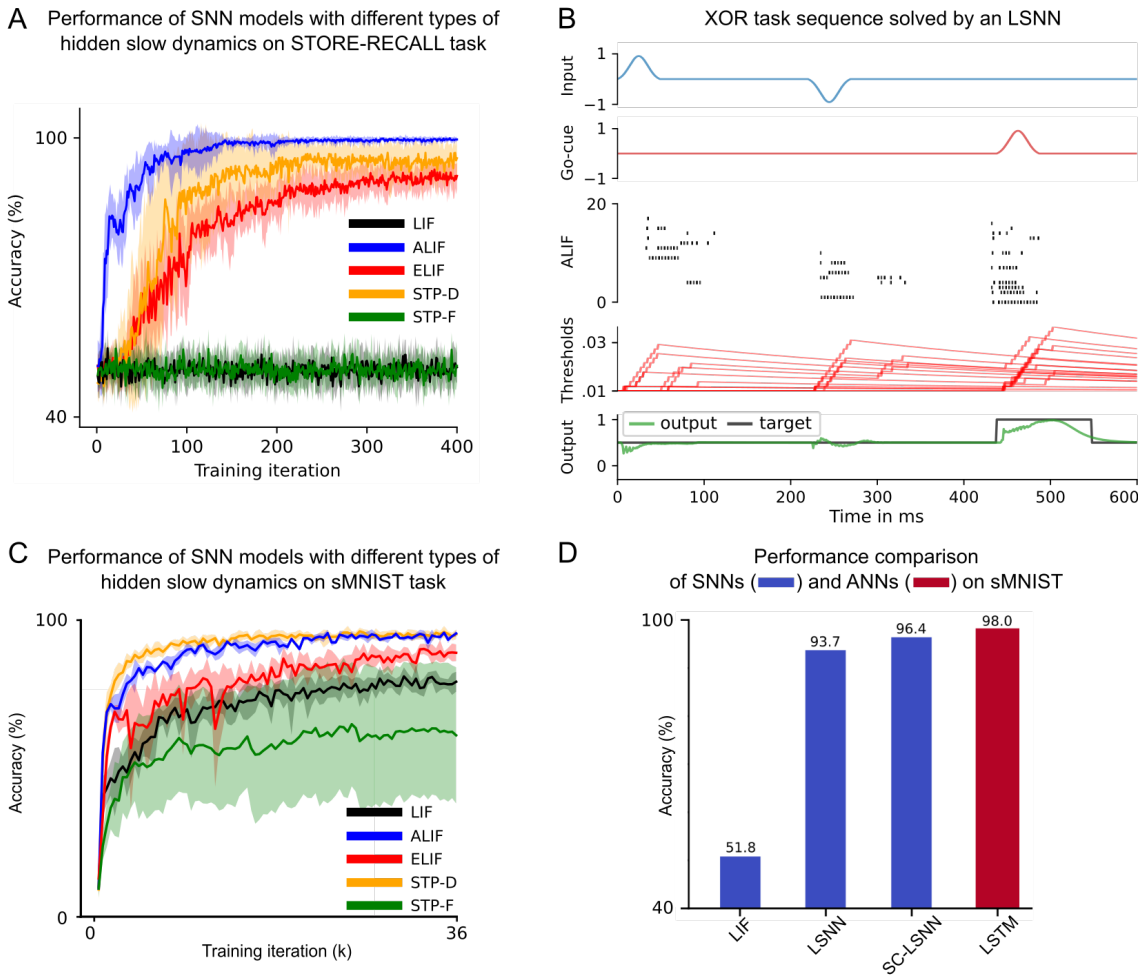

Figure 3: Performance comparisons for common benchmark tasks that require substantial integration of information over time. (**A**) Learning curves of networks with different slow processes, for a 1D version of the STORE-RECALL task from Fig. 2. The standard SNN ("LIF") as well as SNNs with STP-F cannot learn the task. SNNs with STP-D come closest to the performance level of the LSNN, but require substantially longer training. Mean accuracy and standard deviation are shown for 7 runs with different network initializations for all 5 network types. (**B**) Trained LSNN solving the delayed-memory XOR task (Huh and Sejnowski, 2018). Plot of a trial with input consisting of two different types of pulses is shown. From top to bottom: Input pulses, go cue, neuron spike raster, threshold traces, network output. (**C**) Learning curves of five variants of the SNN model (same as in **A**) for the sMNIST time series classification task. Mean accuracy and standard deviation are shown for a minimum of 4 runs with different network initializations for all 5 network types. (**D**) sMNIST performance of two versions of LSNNs are compared with that of an equally large network of LIF neurons, and an LSTM network. SC-LSNN is a sparsely connected LSNN consisting of excitatory and inhibitory neurons.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

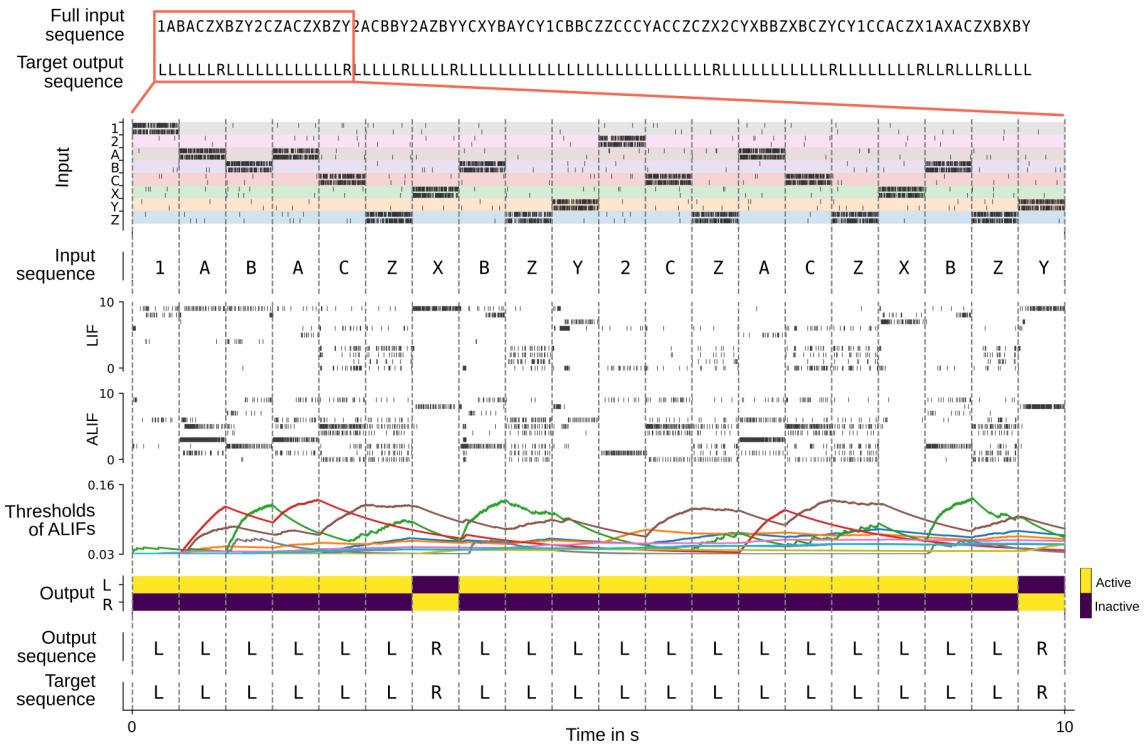

Figure 4: Solving the 12AX task by a network of spiking neurons (LSNN), trained with *random e-prop*. A sample trial of the trained network is shown. From top to bottom: Full input and target output sequence for a trial, consisting of 90 symbols each, blow-up for a subsequence of the input symbols, firing activity of 10 sample LIF neurons and 10 sample ALIF neurons from the LSNN, time course of the firing thresholds of these 10 ALIF neurons, activation of the two readout neurons, the resulting sequence of output symbols which the network produced, and the target output sequence.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

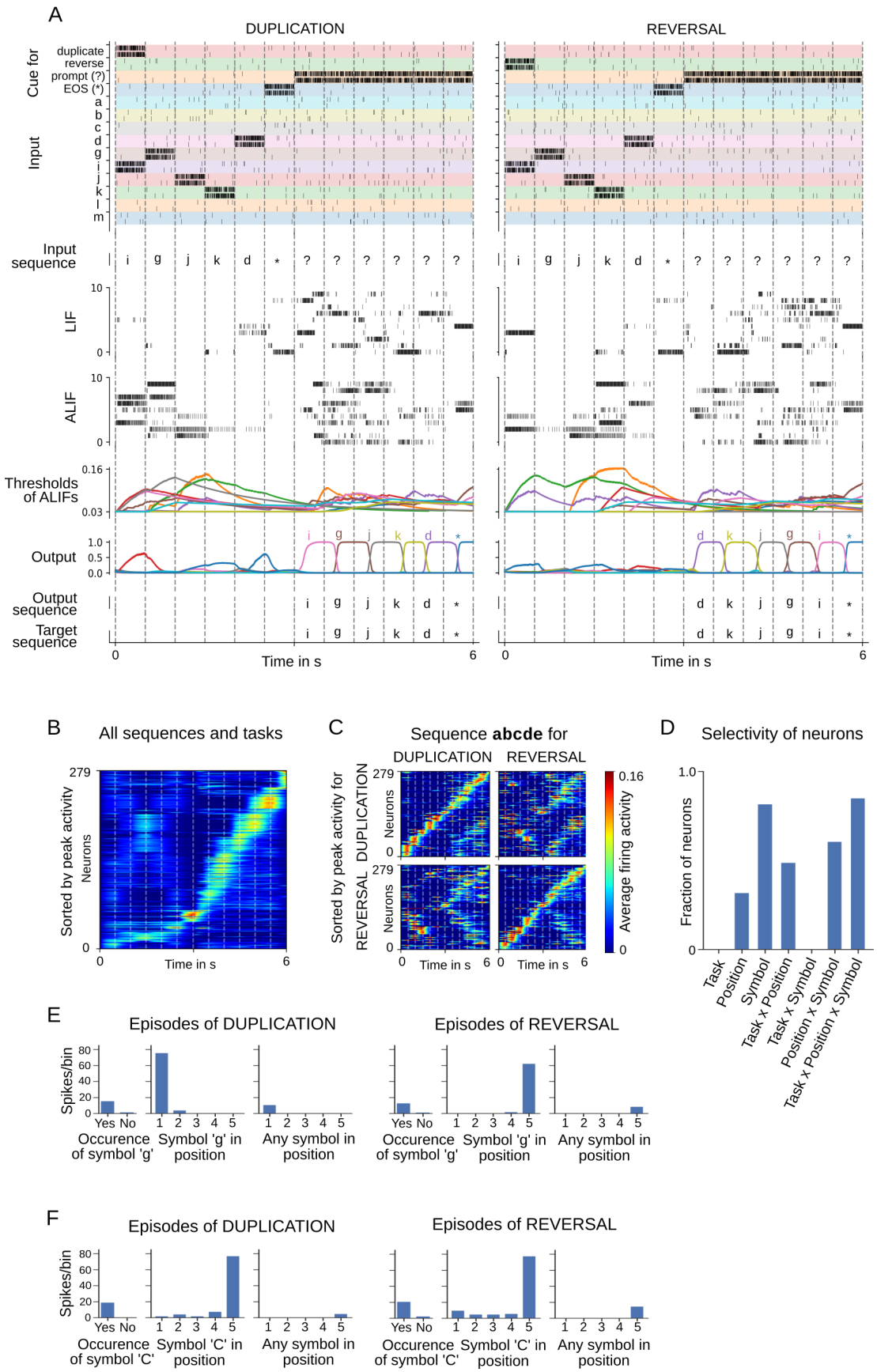

Figure 5: (Caption on the next page.)

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Figure 5: Analysis of an LSNN trained to carry out operations on sequences. (**A**) Two sample episodes where the LSNN carried out sequence duplication (left) and reversal (right). Top to bottom: Spike inputs to the network (subset), sequence of symbols they encode, spike activity of 10 sample LIF, and ALIF neurons in the LSNN, firing threshold dynamics for these 10 ALIF neurons, activation of linear readout neurons, output sequence produced by applying argmax to them, target output sequence. (**B**-**F**) Emergent neural coding of 279 neurons in the LSNN. Neurons sorted by time of peak activity. (**B**) A substantial number of neurons are sensitive to the generic timing of the tasks, especially for the second half of trials when the output sequence is produced. (**C**) Neurons separately sorted for duplication episodes (left column) and reversal episodes (right column). Many neurons respond to input symbols according to their serial position, but differentially for different tasks. (**D**) Histogram of neurons categorized according to conditions with statistically significant effect (3-way ANOVA). Firing activity of a sample neuron that fires primarily when: (**E**) the symbol "g" is to be written at the beginning of the output sequence. The activity of this neuron depends on the task context during the input period; (**F**) the symbol "C" occurs in position 5 in the input, irrespective of the task context.

bioRxiv preprint doi: https://doi.org/10.1101/2020.05.11.081513; this version posted May 12, 2020. The copyright holder for this preprint (which was not certified by peer review) is the author/funder. All rights reserved. No reuse allowed without permission.

Model	test accuracy (%)
FastGRNN-LSQ (Kusupati et al., 2018)	93.18
LSNN	91.21
LIF network	89.04

Table 1: Google Speech Commands. Accuracy of the spiking network models on the test set compared to the state-of-the-art artificial recurrent model reported in (Kusupati et al., 2018). Accuracy of the best out of 5 simulations for LSNNs and LIF networks is reported.