Self-Organized Artificial Grammar Learning in Spiking Neural Networks

Renato Duarte (renato.duarte@bcf.uni-freiburg.de)
Bernstein Center Freiburg, Albert-Ludwigs University
Freiburg im Breisgau, 79104 Germany & Institute for Adaptive and Neural Computation,
School of Informatics, University of Edinburgh, EH8 9AB, United Kingdom

Peggy Seriès (pseries@inf.ed.ac.uk)
Institute for Adaptive and Neural Computation,
School of Informatics, University of Edinburgh, EH8 9AB, United Kingdom

Abigail Morrison (morrison@fz-juelich.de)
Bernstein Center Freiburg, Albert-Ludwigs University
Freiburg im Breisgau, 79104 Germany & Institute of Neuroscience and Medicine (INM-6),
Computational and Systems Neuroscience, Jülich Research Center, 52425, Germany &
Institute of Cognitive Neuroscience, Faculty of Psychology, Ruhr-University Bochum, 44801, Germany

Abstract

The Artificial Grammar Learning (AGL) paradigm provides a means to study the nature of syntactic processing and implicit sequence learning. With mere exposure and without performance feedback, human beings implicitly acquire knowledge about the structural regularities implemented by complex rule systems. We investigate to which extent a generic cortical microcircuit model can support formally explicit symbolic computations, instantiated by the same grammars used in the human AGL literature and how a functional network emerges, in a self-organized manner, from exposure to this type of data. We use a concrete implementation of an input-driven recurrent network composed of noisy, spiking neurons, built according to the reservoir computing framework and dynamically shaped by a variety of synaptic and intrinsic plasticity mechanisms operating concomitantly. We show that, when shaped by plasticity, these models are capable of acquiring the structure of a simple grammar. When asked to judge string legality (in a manner similar to human subjects), the networks perform at a qualitatively comparable level.

Keywords: Sequence Learning; Self-Organization; Plasticity; Artificial Grammar Learning;

Introduction

Sequential organization is a ubiquitous facet of adaptive cognition and behavior. Many of our more fundamental abilities reflect some form of adaptation to the structural regularities of sensory events, as they unfold over time, and the extraction and use of such regularities.

In order to adequately navigate complex, dynamic environments, an agent ought to be able to represent and process sequences of information, use this information in a predictive context, to make inferences about what will happen next, when it will happen and how to react to it, and assemble elementary responses into novel action sequences.

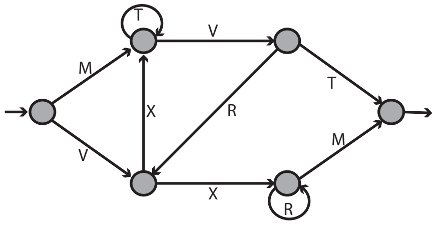
It is thus of central importance to elucidate how knowledge about sequential structure is acquired, represented in memory and expressed in behavior, and to understand the nature and characteristics of such knowledge representations and of the underlying acquisition mechanisms. Importantly, such pursuit must be grounded by the biophysical properties of the neural processing infrastructure. Mapping such complex computational processes to the underlying neuronal processes and assessing the properties of the neuronal system responsible for their implementation is not straightforward, but it is likely to yield important insights into the nature of neural computation.

Artificial Grammar Learning

The problem of sequence learning has a long tradition in cognitive science and psycholinguistic research. Considerable effort has been devoted to the question of whether and under which conditions, the acquisition of complex, rule-governed knowledge can be performed in an incidental or implicit manner, i.e., “without any requirements of awareness of either the process or the product of acquisition”(A. S. Reber, Walkenfeld, & Hernstadt, 1991). These studies exploit the fact that our ability to deal with complex sequential structure is most evident in language acquisition and processing, transforming the problem of sequence learning into the largely equivalent problem of grammar learning, which can be addressed within the domain of language syntax. In fact, growing evidence suggests that language acquisition and processing is mediated by implicit sequence learning and structured sequence processing (K. M. Petersson & Hagoort, 2010), thus involving common mechanisms.

A typical AGL experiment consists of a learning or acquisition phase and a test phase. During acquisition, participants are exposed to a set of symbol sequences generated from a formal grammar (a complex rule system, whose rules can be described by the allowed transitions of a directed graph, e.g. Figure 1), often in the form of a short-term memory task. During the subsequent test phase, subjects are informed about the existence of an underlying set of rules and instructed to classify a novel set of sequences as grammatical or not, based on their immediate intuitive judgement.

A robust and well replicated finding is that subjects perform significantly above chance, and performance improves if subjects are exposed to multiple sessions of implicit acquisition. This means that humans are able to acquire knowledge



MTTVT
VXRRRRM
VXVRXRRM
MTTTTVRXRRM
VXTVRXRRM
VXTTTTTTVT
MTVT
MTTVRXM
VXVRXRRM

Figure 1: Representation of a grammar commonly used in AGL studies (the Reber Grammar (A. Reber, 1967)) alongside some examples of strings it generates.

about complex rule systems by mere exposure and without any performance feedback. The neurobiological correlates of such ability are just starting to be unravelled and the precise neurocomputational implementation is still largely unexplored.

Computing with neural circuits

While the myriad of complex features one encounters in neurobiology hinders a detailed description and understanding of all the system's components and their interactions, a certain degree of universality in both structure and dynamics can be encountered in the neocortical microcircuitry, and ought to be exploited in order to build simplified models capturing the essential features that are likely to be most relevant for the computations it performs.

One such property is the ubiquitous recurrent connectivity, making Recurrent Neural Network (RNN) models suitable candidates for neurocomputational studies. The approach used throughout this study is inspired by the *Reservoir Computing* framework (Lukoševicius & Jaeger, 2009), whose main strength lies in its simplicity. The idea is to provide a large enough neuronal pool with a time-varying input and train a set of readout units (receiving convergent synaptic input from the pool) to perform some spatio-temporal transformation on the input stream based on the high-dimensional representation generated by the network activity. The neurons in the pool (or reservoir) are randomly, sparsely and recurrently connected and can comprise varying degrees of biological detail. Apart from being simple to implement and train, these models have been consistently shown to be very computationally powerful and readouts from such circuits can be trained to perform meaningful, non-trivial computations.

The main caveat of this approach when it comes to brain-inspired modelling, lies in the randomness of the recurrent connections, which remain fixed and untrained. Biological networks are known to be shaped by several adaptation mechanisms, making them effective processing devices and it is reasonable to assume that the nature of the processing task will differentially tune the circuit's properties to the current needs.

Self-Organizing Spiking Network

We explore a simplified spiking network model (Zheng, Dimitrakakis, & Triesch, 2013), whose connectivity structure and spiking dynamics are shaped by a combination of different plasticity mechanisms, inspired by biology and capable of accurately reproducing the statistics and fluctuations of synaptic connectivity patterns encountered in the neocortex, while actively maintaining a stable firing activity.

The model consists of a reservoir of noisy, threshold spiking neurons. The network is subdivided in excitatory and inhibitory populations (N^E excitatory and $N^I = 0.2 \times N^E$ inhibitory neurons). All synaptic connections onto the excitatory population are subjected to adaptation through synaptic plasticity (see below), whereas connections to inhibitory neurons are fixed and static. W^{EE} and W^{EI} are sparse, with connection probabilities of $p_{EE} = 0.1$ and $p_{EI} = 0.2$, respectively, whereas W^{IE} is full (all-to-all connectivity). No synapses among inhibitory neurons and no 'autapses' are allowed. The initial synaptic strengths are randomly drawn from a uniform distribution in $[0,1]$ and subsequently normalized such that the total synaptic input each neuron receives sums up to 1.

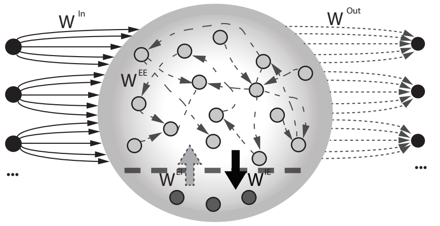


Figure 2: Schematic representation of the network used in this study. Dashed arrows represent dynamic synapses.

The network's activity, at discrete time t is given by the binary vectors $x^E(t) \in \{0,1\}^{N^E}$ and $x^I(t) \in \{0,1\}^{N^I}$, whose dynamics are described by:

$$x_i^E(t+1) = \Theta\left(\sum_{j=1}^{N^E} W_{ij}^{EE} x_j^E(t) - \sum_{k=1}^{N^I} W_{ik}^{EI} x_k^I(t) \right. \\ \left. + \sum_{j=1}^{N^U} W_{ij}^{in} u_j(t) + \xi_i^E(t) - T^E(t)\right) \qquad (1)$$

$$x_i^I(t+1) = \Theta\left(\sum_{j=1}^{N^E} W_{ij}^{IE} x_j^E(t) + \xi_i^I(t) - T^I\right) \qquad (2)$$

The neurons' thresholds (T^E and T^I) are random values initially drawn from a uniform distribution in the interval $[0, T_{max}^E]$ and $[0, T_{max}^I]$, for excitatory and inhibitory units, respectively. The Heaviside step function $\Theta(.)$ constrains the network activations to a binary representation, i.e., a neuron fires if the total drive it receives exceeds its threshold, otherwise it stays silent. ξ^E and ξ^I introduce a small amount of Gaussian white noise with $\mu_\xi = 0$ and $\sigma_\xi^2 = 0.04$.

The readout neuron's output is given by:

$$z_i(t) = \sum_{i=1}^{N^O} W^{Out}_{ik} x^E_k(t) \tag{3}$$

The only form of supervised learning is performed to obtain W^{Out}, the synapses from the reservoir to the readout units. The learning algorithm we chose to use was the FORCE algorithm (Sussillo & Abbott, 2009), which is a recursive algorithm, involving online, error-directed synaptic changes. The weights onto the readout neurons are updated according to:

$$\Delta W^{out} = -P(t)e(t)x^E(t) \tag{4}$$

where the $N^E \times N^E$ matrix $P(t)$, contains a running estimate of the inverse of the correlation matrix of $x^E(t)$.

Adaptation

The most important aspect of this model is the fact that its structure and dynamics are shaped by the synergistic combination of several plasticity mechanisms (see (Zheng et al., 2013) for a more complete description of the model and plasticity rules used in this study):

Spike-Timing-Dependent Plasticity The most well-known form of activity-dependent synaptic plasticity relies on its dependence on pre- and postsynaptic spike times. To implement spike-timing-dependent synaptic modifications, all the synapses onto excitatory neurons are altered by small, fixed amounts ($\eta = 0.001$), according to:

$$\Delta W^{EE}_{ij}(t) = \eta(x^E_i(t)x^E_j(t-1) - x^E_i(t-1)x^E_j(t)) \tag{5}$$

In order to balance the excitatory and inhibitory inputs to the excitatory neurons, a form of inhibitory STDP is also introduced modifying the weights of inhibitory synapses as:

$$\Delta W^{EI}_{ij}(t) = -\eta x^I_j(t-1)\left(1 - x^E_i(t)(1 + \frac{1}{\mu IP})\right) \tag{6}$$

By maintaining the balance and stabilizing the firing rate, this rule also serves a homeostatic purpose.

Intrinsic Plasticity Distributes the network activity evenly throughout the excitatory neurons, by regulating their responsiveness and enforcing the maintenance of a constant average firing rate. The threshold of a neuron that has just been active is decreased, while the threshold of an inactive neuron is increased by the same amount ($\eta_{IP} = 0.001$):

$$T^E_i(t+1) = T^E_i(t) + \eta_{IP}(x^E_i(t) - H^{IP}_i) \tag{7}$$

where $H^{IP} \sim \mathcal{N}(\mu_{IP}, \sigma^2_{IP})$ sets the target firing rates.

Synaptic Normalization Proportionally distributes the strengths of incoming synapses onto an excitatory neuron ($W^{E\alpha}$, with $\alpha \in \{E, I\}$). This rule keeps the weights bounded, while maintaining their relative distribution, thus implementing a form of competition among synapses:

$$W^{E\alpha}_{ij}(t)/\sum_j W^{E\alpha}_{ij}(t) = W^{E\alpha}_{ij}(t) \tag{8}$$

Structural Plasticity With a small probability $p_c = 0.001$, a new synapse, with a strength of 0.001 is added between a random pair of excitatory neurons.

Input Structure

The input consists of symbolic temporal sequences ($S_t = \sigma_1, \sigma_2, ..., \sigma_T$), whose symbols σ_i are drawn from the finite alphabet $\mathcal{A} = \{\#, M, V, R, T, X\}$, following the set of rules specified by the Reber Grammar (RG) (traversing the allowed transitions of the graph depicted in Figure 1) and concatenating the strings it generates with the symbol #, thus allowing the generation of potentially infinite symbol sequences.

An input layer (u) consisting of N^U neurons is created, along with a (full) matrix of connection weights (W^{in}), whose values are uniformly drawn from $[-1, 1]$, connecting the input layer to the main reservoir. W^{in} is $N^U \times N$ dimensional, N being the number of neurons in the main reservoir and $N^U = |\mathcal{A}|$. Sequence symbols are encoded in the input layer (u) with a *one-hot* or *exclusive* coding scheme, i.e., all activities are set to 0, except the one corresponding to the current symbol, which is set to 1.

Predictive Modelling for the RG

In order to assess the network's ability to process the symbolic sequences and to extract the underlying regularities, we train it to perform 1-step prediction. Given the nature of the grammar, predictions are not deterministic so, if the network is able to acquire the relevant representations from the training data, it should activate all the output neurons corresponding to allowed transitions from the current input. Hence, it becomes appropriate for the prediction to take the form of a probability distribution of possible next items, and so the readout output is rectified (all negative values set to 0) and normalized ($\hat{P}_i(t) = z_i(t)/\sum_{j=1}^{N^O} z_j(t)$).

To adequately quantify the network's performance, these probability estimates ought to be compared with some context-dependent likelihood vector for the desired output, i.e., a target probability distribution for the possible next symbols (P_{target}). To be able to equate our results with human subjects, we must assume that the network is a naive learner, and that the ground true probabilities are not accessible. The learner is exposed to a training sequence and the underlying goal is to learn a model $\hat{P}$ that assigns some probability to future outcomes, given some context. We start by evaluating the performance of a series of general-purpose prediction algorithms under the same conditions, while accounting for different context lengths. The idea is that, if the network builds an appropriate internal model of the data, its predictions should be as good as the best performing model.This way, we can also disambiguate the amount of contextual information that the network should memorize in order to make accurate predictions, which can be fixed (*N-gram Markov Models*) or vary based on the locally available statistics of the training data (*Variable Order Markov Models* (VMMs)).

The literature on VMMs is quite extensive, so we restricted our analysis to some well-known algorithms that have been tested in discrete sequence prediction tasks whose nature resembled that of our symbolic sequences (Begleiter, El-Yaniv, & Yona, 2004). The prediction performance of these algorithms is evaluated using the average log-loss $l(\hat{P}, S_{Test})$ of the model, with respect to the test sequence $S_{Test} = \sigma_1 ... \sigma_T$:

$$l(\hat{P}, S_{Test}) = -\frac{1}{T}\sum_{i=1}^{T} \hat{P}(\sigma_i | \sigma_1 ... \sigma_{i-1}) \tag{9}$$

The average log-loss is equivalent to the likelihood $\hat{P}(S_{Test}) = \prod_{i=1}^{T} \hat{P}(\sigma_i | \sigma_1 ... \sigma_{i-1})$ and minimizing the average log-loss is equivalent to maximizing a probability assignment for the entire test sequence. The results of this preliminary analysis are provided in Table 1 (see (Begleiter et al., 2004; Dimitrakakis, 2010) for a complete description of the algorithms). All the models were trained on the same training sequence (length of 1×10^5 symbols) and tested on the same test sequence (5×10^4 symbols). Whenever the models contained some hyperparameter that required tuning (such as maximum context depth), these were determined by 10-fold cross-validation over the training data.

Table 1: Prediction algorithms tested. The acronyms in the variable context models stand for: Lempel-Ziv (LZ), Decomposed Context Tree Weighting (D-CTW), Binary Context Tree Weighting (Bi-CTW), Prediction by Partial Match (PPM) and Bayesian Variable Order Markov Model (BVMM).

Fixed Context		**Variable Context**	
Algorithm	Log-Loss	Algorithm	Log-Loss
1-Gram	2.5069	LZ	1.2753
2-Gram	1.6702	D-CTW	1.0801
3-Gram	1.0673	Bi-CTW	1.0882
4-Gram	1.0678	PPM	1.0971
5-Gram	1.0679	BVMM	1.0677

The results showed that a $3-gram$ model is the best model for this data, so the probability estimates given by this model were set as the target (i.e., $P_{target} = P(S_t = \sigma_i | S_{t-2}, S_{t-1})$), a result consistent with a careful inspection and analysis of the RG (K. Petersson, Grenholm, & Forkstam, 2005).

Prediction Performance

After setting the target probabilities, we need to compare them with the estimates provided by the readout output ($\hat{P}$). For that purpose, we measure the Kullback-Leibler divergence, which can be interpreted as a measure of the information lost when $\hat{P}$ is used to approximate P_{target}:

$$D_{KL}(P_{target} || \hat{P}) = \sum_{i \in \mathcal{A}} P^i_{target} \times log\left(\frac{P^i_{target}}{\hat{P}^i}\right) \tag{10}$$

To adequately compare the results obtained from different network implementations, we determine prediction performance as the mean normalized D_{KL} ($Perf ormance = -\frac{1}{T}\sum_{t=1}^{T} 1 - exp(-D_{KL}(t))$), which results in a value of 1 if $D_{KL} = 0$. The results depicted in Figure 3 were obtained by training networks of different sizes (N^E) with an increasing number of strings and subsequently testing them on a fixed size test corpus composed of 1×10^4 strings. The benefits of self-organization through plasticity are evident. Given enough training data (> 1000 strings), even a small network of only 200 neurons is capable of adequately representing the temporal relations in the input sequence and predict the test sequence with a significantly high performance (close to 0.8, reflecting a very good agreement between the output estimates and the target distributions), despite the non-trivial complexity of the task, requiring the maintenance of context information in the neuron's activities. Static networks, on the other hand perform barely above chance level, even when large networks are trained with large amounts data.

Furthermore, we determine the stability of the solution found by the readout algorithm by measuring its norm, the rationale for doing so being that a large value of $|W^{Out}|$ corresponds to an output which depends heavily on some dimensions of state space, while ignoring others, thus leading to an unstable output which is very sensitive to variations in the activity of some neurons and does not accurately reflect the population dynamics. The results demonstrate that the static networks achieve the higher performances in conditions when the solutions are less stable, whereas in the plastic case, the global mean squared error of the readout output is very small and the solutions found are stable, thus accurately portraying the network dynamics.

Additionally, the spiking activity within the reservoir, while processing the input sequence, retains a healthy dynamics (see Figure 4), with asynchronous (low pairwise correlations) and irregular (coefficient of variation of the inter-spike intervals close to 1) firing activity. The average population firing rate is also actively maintained within the desired value, with small fluctuations around the mean.

In order to obtain a better understanding of the impact of plasticity in shaping the network activity and it's ability to adequately learn the input structure, we systematically 'knocked out' each plasticity mechanism individually and assessed how much this would impact performance. The results depicted in Figure 5 (top) show that while structural plasticity (SP) is the less relevant for the network's ability to perform the task, synaptic normalization (SN) and inhibitory STDP (iSTDP) have a particularly large influence. If we analyse how activity spreads from one time step to the next, as displayed in Fig-

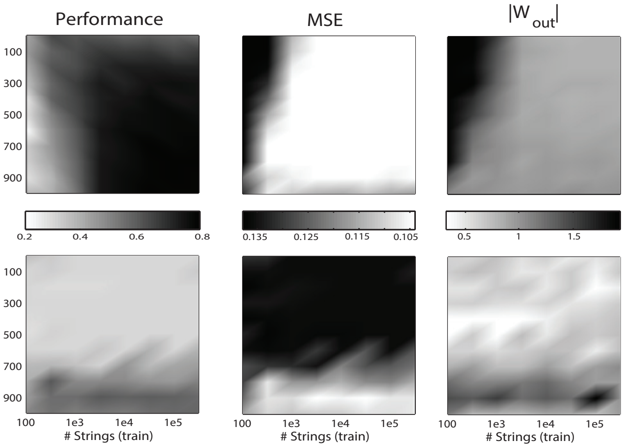


Figure 3: Performance of dynamic (top row) and static networks (bottom row), for increasing amounts of training and different network sizes. Results depict the average of 10 simulations per condition.

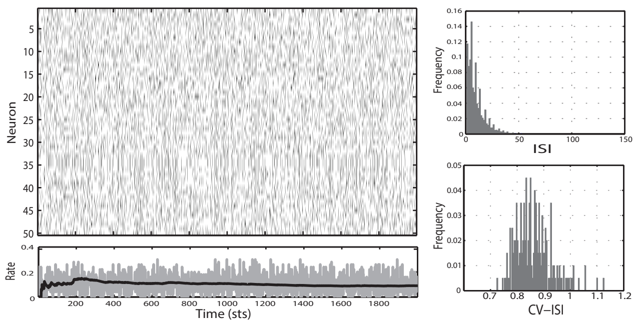


Figure 4: Snapshot of network activity in the initial steps of the test phase. The figure depicts the activity of 50 randomly selected excitatory neurons as well as the distributions of inter-spike intervals and their coefficient of variation.

ure 5 (bottom), a pattern distinctly emerges. The combined action of all the plasticity mechanisms, imposes a contractive dynamics in the network's state space, important to create adequate trajectories that can be easily readout. If either of the mechanisms with a bigger impact on performance is taken out, the activity spreads and the network is allowed to explore a vaster region of state space, which has a natural deleterious effect on the separation of the input into distinct trajectories and hinders the network's ability to acquire and reflect the structure of its input. Besides, the spiking activity becomes more regular and synchronized (insets in Figure 5).

Judging String Legality

To complement the analysis, we devised a protocol to classify string sets, in a manner that can be equated and compared to human behavioral experiments. The network is thus expected to discern whether a string was generated by the grammar and adheres to its rules or not (Grammatical (G) or Non-Grammatical (NG)). Following the same line as in the previous section, we interpret the network's output as a probability distribution, but now we store, on each step, the probability mass that the readout assigned to the correct symbol, the one that actually occurs. We then take the product of all symbol predictions on an individual string and normalize it by taking the logarithm of this value and dividing it by the string length. Formally, consider a string L of length N and the network's next-symbol predictions $\hat{P}$, the string 'likelihood ratio' (LR) is given by:

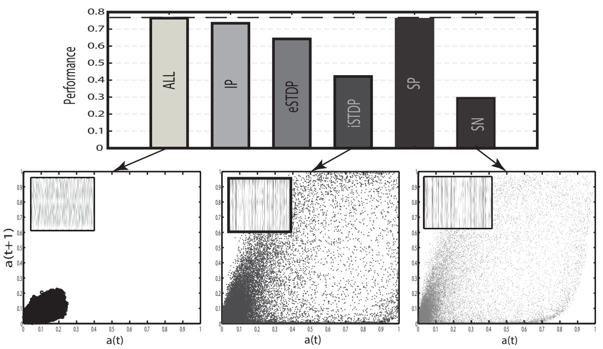


Figure 5: Impact of the different plasticity rules on prediction performance and network dynamics. Top: Performance results obtained with all mechanisms and in the absence of those highlighted. Bottom: Spread of activity from one time step to the next, in a network with all plasticity mechanisms (left), without iSTDP (middle) and without SN (right).

$$LR_L = -\frac{1}{N-1} log_2(\prod_{t=1}^{N-1} \hat{p}_i(t)), \qquad (11)$$

where $\hat{p}_i(t) = \hat{P}(L(t+1) = \sigma_i)$, knowing that $L(t+1) = \sigma_i$.

We treat each network simulation as a subject, train and test them using the exact same string set used in a human AGL experiment (for a detailed description of the experimental methods used for the acquisition of the behavioral data, including the characteristics of the stimulus material, see (Forkstam, Hagoort, Fernandez, Ingvar, & Petersson, 2006) and references therein). Apart from grammaticality status, the test strings are also divided in subsets that account for how frequently 2 and 3 letter chunks making up the strings appear in the training set (low (L) and high (H) associative chunk strength). Human classification performance is assessed in terms of endorsement rates (i.e. fraction of strings classified as grammatical) and compared with the network likelihood ratios, averaged over all the strings of each type and all the 'experimental subjects' (Figure 6). The results of this analysis display a similar pattern of variation from the untrained (baseline) to the trained (grammaticality) conditions between the network's normalized likelihood ratio scores and the human endorsement rates, in all conditions and for the different string types. Similar to human subjects, the networks are highly sensitive to grammaticality status and, after a training or acquisition phase, there is a remarkable difference between G/NG items in both NLR scores and endorsement rates.

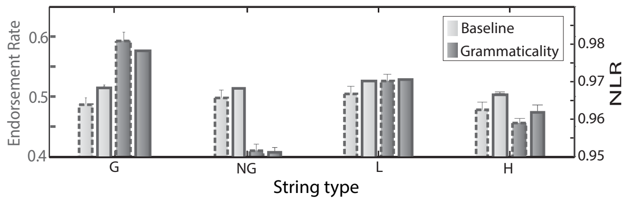


Figure 6: Grammaticality classification. Comparison of the network's likelihood ratio (bars enclosed by full lines, right axis) with human endorsement rates (bars enclosed by dashed lines, left axis), before and after training.

Discussion

In this work, we have showed that, through the process of self-organization, mediated by biologically plausible plasticity mechanisms, spiking neural networks are capable of acquiring the structure of a simple (regular) grammar and developing a reliable predictive model, whose predictions are comparable to those of the best performing learning algorithms. By mere exposure to strings generated by the grammar, the network functionally develops stable, confined trajectories that can be accurately mapped, readout and used to make correct, context-dependent predictions. Furthermore, the magnitude of plasticity-induced changes that the network is subjected to tends to stabilize, allowing it to develop stable and healthy dynamics throughout training and testing, despite having some of its components permanently subjected to adaptation. When asked to judge string legality, based on a likelihood metric computed on the network's output estimations, it performs at a qualitatively comparable level to that of human subjects on a similar task, displaying a particularly high sensitivity to the grammaticality status of the tested strings.

While the focus of this work has been on human performance data as reported in the psycholinguistic literature, it should be noted that, from a purely computational perspective, the task of learning a finite-state (regular) language, such as that generated by the RG, bears only a tenuous similarity to human language learning, which is known to require much more complex dependencies. Previous work employing simpler models, such as the Simple Recurrent Network (SRN), has already accounted for the ability of such learning devices to acquire the dependencies present in such languages (K. Petersson et al., 2005) and even extended these results to more informative classes (e.g. (Elman & Diego, 1991)). However, most of these previous studies shows little or no parallel with the biophysical properties of the neuronal system both in terms of structure (typically employing fully connected recurrent networks, with real-valued, continuous unit activations) and learning mechanisms (based on some form of error backpropagation). What sets the current approach apart from previous work is the close parallel with biology at various levels, from the single neurons' activations to the unsupervised learning mechanisms shaping population dynamics. Whether the present model is computationally capable of coping with the high amount of structural complexity attributed to natural language in a formal sense (of which we are just skimming the surface), remains an open question to be addressed in future work.

Acknowledgments

We thank K. M. Petersson for insightful comments and for providing the behavioral data as well as the anonymous reviewers of the initial submission for the useful comments provided.This work was partially funded by the Erasmus Mundus Joint Doctoral Program EuroSPIN, BMBF Grant 01GQ0420 to BCCN Freiburg, the Junior Professor Program of Baden-Württemberg, the Helmholtz Alliance on Systems Biology (Germany), the Initiative and Networking Fund of the Helmholtz Association and the Helmholtz Portfolio theme "Supercomputing and Modelling for the Human Brain".

References

Begleiter, R., El-Yaniv, R., & Yona, G. (2004). On prediction using variable order Markov models. *Journal of Artificial Intelligence Research*, *22*, 385–421.

Dimitrakakis, C. (2010). Bayesian variable order Markov models. In *Proceedings of the 13th international conference on artificial intelligence and statistics* (pp. 161–168).

Elman, J. L., & Diego, S. (1991, September). Distributed representations, simple recurrent networks, and grammatical structure. *Machine Learning*, *7*(2-3), 195–225.

Forkstam, C., Hagoort, P., Fernandez, G., Ingvar, M., & Petersson, K. M. (2006). Neural correlates of artificial syntactic structure classification. *NeuroImage*, *32*(2), 956–67.

Lukoševicius, M., & Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, *3*(3), 127–149.

Petersson, K., Grenholm, P., & Forkstam, C. (2005). Artificial grammar learning and neural networks. In *Proceedings of the cognitive science society.*

Petersson, K. M., & Hagoort, P. (2010). The neurobiology of syntax: beyond string sets. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, *367*(1598), 1971–83.

Reber, A. (1967). Implicit learning of artificial grammars. *Journal of verbal learning and verbal behavior*, *6*, 855–863.

Reber, A. S., Walkenfeld, F. F., & Hernstadt, R. (1991). Implicit and explicit learning: individual differences and IQ. *Journal of experimental psychology. Learning, memory, and cognition*, *17*(5), 888–96.

Sussillo, D., & Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron*, *63*(4), 544–557.

Zheng, P., Dimitrakakis, C., & Triesch, J. (2013). Network self-organization explains the statistics and dynamics of synaptic connection strengths in cortex. *PLoS computational biology*, *9*(1), e1002848.