S P A R S I T Y T H R O U G H E V O L U T I O N A R Y P R U N I N G P R E V E N T S N E U R O N A L N E T W O R K S F R O M O V E R F I T T I N G

A P R E P R I N T

R i c h a r d C. G e r u m
Biophysics Group, Department of Physics
Friedrich Alexander University Erlangen-Nürnberg (FAU), Germany

A n d r é E r p e n b e c k
The Raymond and Beverley Sackler Center for Computational Molecular and Materials Science,
School of Chemistry, Tel Aviv University (TAU), Israel

P a t r i c k K r a u s s
Experimental Otolaryngology,
Neuroscience Lab, University Hospital Erlangen, Germany
Cognitive Computational Neuroscience Group
at the Chair of English Philology and Linguistics,
Friedrich-Alexander University Erlangen-Nürnberg (FAU), Germany

A c h i m S c h i l l i n g
Experimental Otolaryngology,
Neuroscience Lab, University Hospital Erlangen, Germany
Cognitive Computational Neuroscience Group
at the Chair of English Philology and Linguistics,
Friedrich-Alexander University Erlangen-Nürnberg (FAU), Germany

D e c e m b e r 3 , 2 0 1 9

C o r r e s p o n d i n g a u t h o r:
Dr. Achim Schilling
Neuroscience Group
Experimental Otolaryngology
Friedrich-Alexander University of Erlangen-Nürnberg
Waldstrasse 1
91054 Erlangen, Germany
Phone: +49 9131 8543853
E-Mail: achim.schilling@uk-erlangen.de

K e y w o r d s  E v o l u t i o n · A r t i f i c i a l N e u r a l N e t w o r k s · M a z e T a s k · E v o l u t i o n a r y A l g o r i t h m · O v e r f i t t i n g · B i o l o g i c a l P l a u s i b i l i t y
ABSTRACT

Modern Machine learning techniques take advantage of the exponentially rising calculation power in new generation processor units. Thus, the number of parameters which are trained to resolve complex tasks was highly increased over the last decades. However, still the networks fail — in contrast to our brain — to develop general intelligence in the sense of being able to solve several complex tasks with only one network architecture. This could be the case because the brain is not a randomly initialized neural network, which has to be trained by simply investing a lot of calculation power, but has from birth some fixed hierarchical structure. To make progress in decoding the structural basis of biological neural networks we here chose a bottom-up approach, where we evolutionarily trained small neural networks in performing a maze task. This simple maze task requires dynamical decision making with delayed rewards. We were able to show that during the evolutionary optimization random severance of connections lead to better generalization performance of the networks compared to fully connected networks. We conclude that sparsity is a central property of neural networks and should be considered for modern Machine learning approaches.
Introduction

Sparsity is a characteristic property of the wiring scheme of the human brain, which consists of about $8.6 \times 10^{10}$ neurons [Herculano-Houzel (2009)], interconnected by approximately $10^{15}$ synapses [Sporns et al. (2005); Hagmann et al. (2008)]. Thus, from almost $10^{22}$ theoretically possible synaptic connections, only one of 10 million connections is actually realized. This extremely sparse distribution of both neural connections and activity patterns is not a unique feature of the human brain [Hagmann et al. (2008)] but can also be found in other vertebrate species such as for example mice and rats [Perin et al. (2011); Oh et al. (2014); Kerr et al. (2005); Song et al. (2005)]. Even evolutionary very old organisms with a quite simple nervous system such as the nematode *C. elegans* with only 302 neurons [Jarrell et al. (2012)] and over 7000 connections [J.G. White, E. Southgate, J.N. Thomson (1986)], show sparsity. Besides the described sparsity of virtually all nervous systems, many biological neural networks also show small world properties [Watts and Strogatz (1998); Amaral et al. (2000); Latora and Marchiori (2001); Bassett and Bullmore (2006), such as scale free connectivity patterns [Perin et al. (2011); van den Heuvel and Yeo (2017); Bassett et al. (2006)].

The described sparsity is the result of both, phylogenetic and ontogenetic adaptations. Even though, almost all species’ immature nervous systems are already very sparse, this sparsity is even further increased during development and maturation of the organisms’ nervous systems [Low and Cheng (2006)]. In fact, the infant human brain contains two times more synapses than the adult brain [Kolb and Gibb (2011)]. Analogously, the immature nervous system of *C. elegans* contains more synapses than the adult form [Oren-Suissa et al. (2016)].

But pruning is not restricted to axons and synaptic connections. It even extends to the total number of neurons, which also decreases during development. For instance, the immature nervous system of *C. elegans* initially consists of 308 neurons [Chalfie (1984)], whereas the adult form contains only 302 neurons [Jarrell et al. (2012)]. And also in humans, the number of neurons decreases during development [Yeo and Gautier (2004)]. These ontogenetic changes are referred to as pruning [Paolicelli et al. (2011)], and it seems to be a universal phenomenon for all species from *C. elegans* to humans. Furthermore, pruning is found to be mandatory for healthy development [Hong et al. (2016)]. In cases where normal synaptic pruning fails, this may even lead to disorders like schizophrenia [Boksa (2012)].

Since sparse connectivity architectures are realized on all scales and in a vast number of organisms of different complexity, it can be assumed that sparse connectivity is a general principle in neural information processing systems, leading to advantages compared to densely connected networks. One major advantage of sparse artificial deep neural networks used for image classification in comparison to fully connected networks, is the reduction of computational costs while at the same time boosting the ability to generalize [Han et al. (2015); Anwar et al. (2017); Wen et al. (2016); Mocanu et al. (2018)].

However, these pure feed forward network architectures show low biological plausibility as they neither have the ability of dealing with time series data, nor have any memory-like features. Efficient processing of time series data in artificial neural networks is a complex task with a bunch of limitations. The technique of training the neural networks by unfolding the data in time is computational expensive and time consuming and leads to effects such as vanishing or exploding gradients [Hochreiter (1998); Pascanu et al. (2012)], which have been partly overcome by the introduction of Long-Short-Term-Memory Networks [Schmidhuber and Hochreiter (1997)]. However, these networks are difficult to interpret from a biological point of view.

To overcome these limitation, novel biologically inspired approaches for processing time series data were introduced called reservoir computing [Verstraeten et al. (2007); Lukoševičius et al. (2012)]. A so called reservoir of neurons with fixed (i.e. not adjusted by training) random recurrent connections is used to calculate higher-order correlations of the input signal which serve as input for a feed forward output layer that is trained with error back-propagation. The properties of the reservoir networks were found to be ideal for biologically inspired parameters with a high sparsity [Alexandre et al. (2009)]. Thus, these reservoirs work best at the edge of chaos, meaning that the parameters have to be chosen so that they are balanced between complete chaos and absolute periodicity [Schrauwen et al. (2007); Bertschinger and Natschläger (2004); Krauss et al. (2019)].

Much effort has been undertaken to apply the technique of reservoir computing on tasks with delayed rewards such as robot navigation in mazes [Antonelo and Schrauwen (2012); Antonelo et al. (2007)].

However, the technique of reservoir computing is still based on the fact that the output layer has to be trained in a supervised way using back-propagation and, thus, complex tasks with a delayed reward are difficult to realize. Thus, in this study we used an evolutionary approach to train networks in solving a maze task. We did not use reinforcement learning approaches as these techniques suffer from the hidden state problem, meaning that not all information is present at any time and from the delayed reward problem [Littman (2001)], which plays an important role in our maze task [Lample and Chaplot (2017)].
Sparsity through evolutionary pruning prevents neuronal networks from overfitting

In our evolutionary system we were able to show that the random severing of connections (evolutionary pruning), without explicitly rewarding sparsity, did lead to a general sparsification of the networks and a better generalization performance.

Figure 1: Example network and mazes. a, Weight matrix of an exemplary network. b, The same network displayed as connections. c, Mazes are 400 cells wide and 22 high. Walls (red) are at all borders and randomly placed in between. The green line depicts one ideal path through the network.

Materials and Methods

Software Resources

All simulation were run on a desktop computer equipped with an i9 extreme processor (Intel) with 10 calculation cores. The complete software was written in Python 3.6 using the libraries sys, os, glob, subprocess, json, natsort, pickle, shutil, NumPy Van Der Walt et al. (2011). Data visualization was done by the use of Matplotlib Hunter (2007) and plots were arranged using the Pylustrator Gerum (2019).

Maze Task

The task for the agents to perform is a maze based on a rectangular grid of 400x22 cells (Fig. 1c). There are two types of cells, free cells and wall cells. Free cells can be entered and wall cells not. The border of the maze consists of walls to prevent agents from leaving the maze. Starting from the left, every 2 to 10 cells a wall with a length between 4 to 20 is inserted. With a probability of 0.25 the wall is inserted from the same side (up or down) as the last wall or with a probability of 0.75 it is inserted from the other side.

Agents start always at the left end of the maze facing to the right. As ‘sensory’ input each agent receives the distance to the wall in front, to the left and to the right (input neurons 0 to 6, cf. Fig. 1a).

If the distance is larger than 10, it is set to 10 (visual range). It also receives the direction, it is currently looking at, as a one-hot encoded, four-neuron input, i.e. one neuron at a time is in state 1 and the others are in state 0. This input serves as a kind of compass. The seven input neurons do exclusively receive input from the environment, but do not get any input from other neurons, thus, they are reset at each time step.

The agent can output three values for the three possible actions: go straight, turn right, or turn left. The action with the highest value is selected (winner takes all) (output neurons 7 to 9, cf. Fig. 1a). When the agent chooses to go straight and the next field is a wall, it is not moved.

After 400 actions, the covered x-distance of the agent is fed to the fitness function which is proportional to the covered distance (cf. Fig. 1c). Thus, the reward is delayed by 400 time steps.
Network

The logic of each agent consists of a fully connected network of \( N = 16 \) neurons, with states \( s \) (cf. Fig. 1a,b). The connection weights \( W \) are initialized with a random value drawn from a uniformly distribution from the interval \([-\sigma, +\sigma]\) with \( \sigma = 4 \cdot \sqrt{\frac{6}{N+\frac{N}{10}}} / 10 \). For each time step \( t \) in the task, the 3+4 input values (distances (left, front, right) and one-hot encoded direction) are set as the states of the neurons #0 to #6. Then one pass thought the network is calculated, \( s_{t+1} = \text{ReLu}(W \cdot s_t) \), with \( \text{ReLu}(x) \) being the rectified linear function

\[
\text{ReLu}(x) = \begin{cases} 
0 & x \leq 0 \\
[x] & x > 0
\end{cases}
\] (1)

The states of the neurons #7 to #9 are used as outputs to choose one of the three possible actions to perform in the maze task (connectivity matrix see Fig. 1h). Thus, the action is determined by a winner-takes-all-method (in the case of no activation, the "move" action is chosen). Our approach is a policy based approach, as the output of the network directly is the action to take and no quality assessment of different states is undertaken.

Evolutionary Algorithm

For optimizing the networks to fulfill the maze task, we use an evolutionary algorithm [Fekiac et al. (2011)]. Therefore, a pool of 1 000 agents is created with a random initialization and 10 mazes are created for the agents to be trained on.

For each iteration, all agents have to perform the maze task and are assigned a fitness, depending on their score in the task. The best half of the agent pool has now the chance to create offspring. The probability to generate an offspring is proportional to their relative fitness compared to the other agents. Agents with a probability of 10% or more are set to a maximal probability of 10% to retain biodiversity. For each of the old agents to be replaced, a parent agent is selected at random according to their reproduction probabilities. Agents can have multiple offspring or no offspring at all.

After offspring generation, each agent is mutated. We used three different mutation types:

- **Weight mutation**: The connection weights \( W \) are each mutated by addition of a Gaussian distributed random variable \( \mu = 0, \sigma = |\sigma_{\text{mut}}| \).
- **Mutation rate change**: The mutation rate \( \sigma_{\text{mut}} \) is also mutated by multiplication with a Gaussian distributed random variable \( \mu = 1, \sigma = \sigma_{\text{mut}} \).
- **Connection mutation**: Existing connections are removed with a probability \( p_{\text{disconnect}} \) and non-existing connections are added with a probability of \( p_{\text{connect}} = p_{\text{disconnect}} \). Removed connections have a weight of 0 and are not subject to weight mutations. Thus, a removed connection cannot be recovered by a simple mutation step, but can only be recovered by a reconnection mutation.

The fitness (eq. 2) is calculated from the squared mean of the square root distances the agent reached in all 10 training mazes (SMR, eq. 3, this is done to favor generalizing agents which perform okay in all mazes against a specializing agent which performs well in only one maze), the maximum mean activation of the neurons and optionally from the mean number of active connections:

\[
\text{fitness} = \text{SMR}_{\text{mazes}}(\text{distances}) - \max_{\text{mazes}}(\text{mean}(\text{activation})) + f_{\text{sparsity}} \cdot \text{connection\_count}
\] (2)

\[
\text{SMR}_{\text{mazes}} = \left( \frac{1}{N_{\text{mazes}}} \sum_{i=1}^{N_{\text{mazes}}} \sqrt{d_i} \right)^2
\] (3)

All experiments are repeated for 5 different seeds for the random number generator that is used for the initial weights and the mutations. The different repetitions were performed on the same 10 training mazes to keep them comparable.

Results

The evolutionary algorithm was able to find solutions enabling the agents to efficiently navigate through the mazes. In all experiments, except the experiment without weight mutations, the agents gradually learned to perform better in the maze tasks over the generations (Fig. 2h-m). The convergence was quite slow, as about 5 000 generations were needed to converge to a stable solution. The convergence behaviour was mostly independent of the seed of the random
Table 1: **Settings for the different experiments.** The initial mutation rate $\sigma_{\text{mut}}$, the probability of removing or restoring a connection $p_{\text{connect}}$ and the sparsity reward factor $f_{\text{sparsity}}$.

| name                                      | $\sigma_{\text{mut}}$ | $p_{\text{connect}}$ | $f_{\text{sparsity}}$ |
|-------------------------------------------|------------------------|-----------------------|------------------------|
| control                                   | 0.01                   | -                     | -                      |
| connection severance                      | 0.01                   | 0.01                  | -                      |
| connection severance no mut               | 0.01                   | 0.001                 | -                      |
| connection severance sparsity reward     | 0.01                   | 0.01                  | 0.1                    |
| sparsity reward                           | 0.01                   | -                     | 0.1                    |

Figure 2: **Performance of networks of the different experiments during training and validation.** a-f. Fitness over generations for different experiments in 10 validation mazes. Curves show 5 different training seeds, evaluated on the same 10 validation mazes. h-m. Fitness during training over generations for the 5 different training seeds. Labels stand for the different properties of the condition: "mut" for mutation of the weights, "sev" for severing/restoring connections, "sprs." for adding a sparsity reward to the fitness function. n-s. Histograms of the connection weights for the different experiments. Zero connections are not included in the histograms. Gray shaded areas in the center indicate which weights can be removed without reducing the fitness.
number generator, except for the "no mutation" condition, which relied strongly on the initial weights. The fitness during training was best for the conditions with no, or low sparsification pressure (Fig. 2h,j,m).

The fitness in validation mazes, that the agents had not seen during training, was more sensitive to the seed than the fitness during training. For the experiments with more sparsification pressure (Fig. 2h,e) the validation fitness did exceed the fitness during training, showing good generalisation, whereas the experiments with lower sparsification pressure (Fig. 2c,f) showed more problems with over-fitting, which means that the fitness is higher during training than during validation.

Validation fitness increased in most runs during training, although some drops in validation fitness were observed, which sometimes recovered after a few generations, but in some cases the validation fitness continued to fluctuate (see Supplementary Material). In general when the mean validation fitness dropped also the variability of the fitness increased, showing that even if some agents found an "over-fitting" solution, it was not quickly adopted by all agents, whereas when the solutions were more general, they seemed more stable and were adopted by the whole pool of agents.

The weight distribution of the final generation was different for each experiment. While the experiments with low sparsification pressure, that also showed overfitting, show more connections and more weights with larger absolute values (Fig. 2p,s) compared to the other experiments that show more small weight values (Fig. 2o,q). Apart from "connection severance no mut" and "connection severance sparsity reward" (48% and 49% negative weights) all experiments show more negative weights (52–55%), referring to inhibitory synapses, a fact which indicates more interesting behaviour and more efficient information processing [Krauss et al. 2019].

In most experiments, the sparsity increased over time (Fig. 3), but in "connection severance" it even slightly decreased after the initial rise and in "sparsity reward" the sparsity fluctuated strongly over time. A higher sparsity was in all cases (except the no mutation case) associated with a higher validation fitness (Fig. 3b), showing that sparsity improves the generalisation behaviour of evolutionary trained networks. A comparison of the training fitness to the validation fitness also shows that for the sparser cases, the training fitness decreases and in contrast to that, the validation fitness increases. Therefore, the sparsification prevents overfitting and enhances the generalisation performance. In addition, it improves computation efficiency in the evolutionary trained networks as the computations, divided between many neurons in the fully connected control group, are, in sparser networks, forced to be carried out on a small subset of neurons.

Furthermore, it could be shown that the networks which perform best in the test mazes develop simple feed-forward structures (cf. Fig. 1h). Additionally, some asymmetry in the connectivity matrix can be observed (cf. Fig. 1b). On the one hand, the bias units prefer the turn towards a certain direction. On the other hand, the connection from the input distance sensor to the turn output (e.g. to turn to the left side $d_{left}$ in example Fig. 1b) is over-represented for one side. Thus, the networks have a preference to go to one side (e.g. turn left, in the example in Fig. 1b), if there is enough space. The bias unit serves as counterpart, if the agent moves along the upper edge (resp. left side seen when moving along the x-axis) and guarantees that the network can walk away from the wall. The simple network architectures allow for the analysis of the functional tasks of certain neurons. This understanding of the functional tasks of neurons in artificial neural networks could potentially help to understand biological neural networks [Jonas and Kording 2017; Kriegeskorte and Douglas 2018]. Different experiments develop quite different solutions to solve the maze task (see Supplementary Material S1). Interestingly, in some experiments similar structures emerge, regardless of the seed. This is especially the case in the "connection severance" experiment, where 4 out of 5 solutions are strikingly similar. This hints at the existence of strongly attractive maxima in the space of possible solutions.

Discussion

In this study we showed that evolutionary pruning of artificial neural networks, evolutionary trained to solve a simple maze task, leads to sparser networks with better generalization properties compared to dense networks trained without pruning. The evolutionary pruning is realized via two mutation mechanisms: First the networks can set a threshold defining the upper limit of the absolute weight value being virtually set to zero. Secondly, some connections are removed (weight set to 0) at random with a given probability. The random removal of existing connections leads to more robust networks with better generalization abilities and thus can be added as additional mechanism to improve the evolutionary training of neural networks.

However, the maze task is still not complicated enough to trigger the development of complex recurrent neural networks which include abilities such as memory. As the organisms in the maze are always provided with a “compass” meaning always know in which direction they are pointing, the task can be solved with a Markov like decision process, as the information of one time step is enough to decide the next action. Thus, after 5000 epochs the best performances are achieved by simple feed forward networks (cf. Fig. 4). Nevertheless, it has to be considered that these networks were not forced to develop feed forward architectures but are a result of the Markov properties of the task they were trained
The here described study does not show a biologically inspired neuron configuration, however, demonstrates that only a small number of neurons is needed to navigate through environment and visualised. The values correspond to the current activation of each node and the colored connections to the weight times decision cannot be made by simply analyzing the current position in the maze Wierstra and Wiering (2004); Littman. Furthermore, the maze task could be extended so that the task is not a simple Markov process, which means that the next movement along the electric gradient Gabel et al. (2007). Thus, the amphid sensory neurons of C. elegans also changes to a random walk) Pierce-Shimomura et al. (1999); Ryu and Samuel (2002). C. elegans shows thermo- and electrotaxis Gabel et al. (2007) which are not simply a biased random walk in contrast to its' chemotaxis (for certain temperatures thermoataxis also changes to a random walk) Pierce-Shimomura et al. (1999); Ryu and Samuel (2002). C. elegans shows a direct movement along the electric gradient Gabel et al. (2007). Thus, the amphid sensory neurons of C. elegans could be seen as and equivalent to the "compass" neurons in our simulation.

The here described study does not show a biologically inspired neuron configuration, however, demonstrates that 16 neurons are enough to perform relatively simple navigation tasks and gives a hint that the development of sparsity has evolutionary advantages. These findings contrast with current developments in the field of artificial intelligence where the size of the networks due to higher calculation power are scaled up Xu et al. (2018).

Furthermore, the maze task could be extended so that the task is not a simple Markov process, which means that the next decision cannot be made by simply analyzing the current position in the maze Wierstra and Wiering (2004); Littman.

Figure 3: Correlation of validation fitness to sparsity and training fitness. a, Sparsity (1 − \( \frac{n_{nonane}}{n_{possible}} \)) as a function of the epochs (generations). b, Correlation of validation fitness to sparsity. Except for the case of "connection severance no mutation" validation fitness increases on average with increased sparsity. c, Correlation of validation fitness to training fitness. Higher training fitness leads in all experiments also to higher validation fitness, indicating that none of the networks runs into severe overfitting.

Figure 4: State of the maze and configuration of the network during a turn. The position and orientation in the maze are denoted by the green triangle. The past trajectory by the green line. Below, the current state of the network is visualised. The values correspond to the current activation of each node and the colored connections to the weight times the activation of the target node (pink negative, green positive). a, Network one step before turn. b, Network encounters a wall and turns. c, Network has just turned and is now heading north. d, Network continues walking straight in the new direction.

on. The simulation shows that simple feed forward networks are able to navigate through a maze using only 16 sparsely connected neurons.
Sparsity through evolutionary pruning prevents neuronal networks from overfitting A PREPRINT

This increase of complexity can be achieved e.g. by removing the compass neurons. Consequently, a neural network which is able to achieve similar performance than networks with compass neurons need to develop some “memory” features. The networks would have to remember which movements they recently executed and they would have to dynamically recall this information. An even more demanding task would be to force the agents to go directly back to the starting point after they passed the maze. This task requires path integration [Wehner and Wehner (1986); Etienne and Jeffery (2004)], which in turn requires the ability to flexibly navigate in physical space like insects [Wehner and Wehner (1986); Müller and Wehner (1988; 1994)]; [Andel and Wehner (2004)], and at least in mammals episodic memory [Etienne et al. (1996); Séguinot et al. (1998); McNaughton et al. (2006)]. It has been demonstrated that these abilities can only be achieved within highly recurrent networks, as they can be found in the hippocampus [Etienne and Jeffery (2004)]. Recently it has been shown that the network architecture of the hippocampus is not limited to spatial navigation, but seems to be domain-general [Aronov et al. (2017); Killian and Buffalo (2018); Nau et al. (2018)] and even allows navigation in abstract high-dimensional cognitive feature spaces [Eichenbaum (2015); Constantinescu et al. (2016); Garvert et al. (2017); Bellmund et al. (2018); Theves et al. (2019)]. Future work will have to investigate, whether networks with the above mentioned abilities can also be found by evolutionary algorithms.

Acknowledgements

This work was supported by the Emergent Talents Initiative (ETI) of the University Erlangen-Nuremberg (grant 2019/2-Phil-01 to PK).

Author Contributions

RG and AS designed the study. RG and AS performed the simulations. RG, AS, AE and PK discussed the results. RG, AS and PK wrote the manuscript.

References

Alexandre, L. A., Embrechts, M. J., and Linton, J. (2009). Benchmarking reservoir computing on time-independent classification tasks. Proceedings of the International Joint Conference on Neural Networks, pages 89–93.

Amaral, L. A., Scala, A., Barthélémy, M., and Stanley, H. E. (2000). Classes of small-world networks. Proceedings of the National Academy of Sciences of the United States of America, 97(21):11149–11152.

Andel, D. and Wehner, R. (2004). Path integration in desert ants, cataglyphis: how to make a homing ant run away from home. Proceedings of the Royal Society of London. Series B: Biological Sciences, 271(1547):1485–1489.

Antonelo, E. and Schrauwen, B. (2012). Learning Slow Features with Reservoir Networks for Biologically-inspired Robot Localization. Neural Networks, 25:178–190.

Antonelo, E. A., Schrauwen, B., and Stroobandt, D. (2007). Event detection and localization for small mobile robots using reservoir computing. In Conference on Artificial Neural Networks, pages 660–669.

Anwar, S., Hwang, K., and Sung, W. (2017). Structured pruning of deep convolutional neural networks. ACM Journal on Emerging Technologies in Computing Systems, 13(3):1–18.

Aronov, D., Nevers, R., and Tank, D. W. (2017). Mapping of a non-spatial dimension by the hippocampal–entorhinal circuit. Nature, 543(7647):719.

Bassett, D. S. and Bullmore, E. (2006). Small-world brain networks. Neuroscientist, 12(6):512–523.

Bassett, D. S., Meyer-Lindenberg, A., Achard, S., Duke, T., and Bullmore, E. (2006). Adaptive reconfiguration of fractal small-world human brain functional networks. Proceedings of the National Academy of Sciences, 103(51):19518–19523.

Bellmund, J. L., Gärdenfors, P., Moser, E. I., and Doeller, C. F. (2018). Navigating cognition: Spatial codes for human thinking. Science, 362(6415):eaat6766.

Bertschinger, N. and Natschläger, T. (2004). Real-time computation at the edge of chaos in recurrent neural networks. Neural Computation, 16(7):1413–1436.

Boksa, P. (2012). Abnormal synaptic pruning in schizophrenia: Urban myth or reality? Journal of psychiatry & neuroscience: JPN, 37(2):75.

Chalfie, M. (1984). Neuronal development in caenorhabditis elegans. Trends in NeuroSciences, 7(6):197–202.
Sparsity through evolutionary pruning prevents neuronal networks from overfitting

Constantinescu, A. O., O’Reilly, J. X., and Behrens, T. E. (2016). Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352(6292):1464–1468.

Eichenbaum, H. (2015). The hippocampus as a cognitive map... of social space. *Neuron*, 87(1):9–11.

Etienne, A. S. and Jeffery, K. J. (2004). Path integration in mammals. *Hippocampus*, 14(2):180–192.

Etienne, A. S., Maurer, R., and Séguinot, V. (1996). Path integration in mammals and its interaction with visual landmarks. *Journal of experimental Biology*, 199(1):201–209.

Fekiac, J., Zelinka, I., and Burguillo, J. C. (2011). A review of methods for encoding neural network topologies in evolutionary computation. *Proceedings - 25th European Conference on Modelling and Simulation, ECMS 2011*, pages 410–416.

Gabel, C. V., Gabel, H., Pavlichin, D., Kao, A., Clark, D. A., and Samuel, A. D. (2007). Neural Circuits Mediate Electrosensory Behavior in Caenorhabditis elegans. *Journal of Neuroscience*, 27(28):7586–7596.

Garvert, M. M., Dolan, R. J., and Behrens, T. E. (2017). A map of abstract relational knowledge in the human hippocampal–entorhinal cortex. *Elife*, 6:e17086.

Gerum, R. (2019). pylustrator: Code generation for reproducible figures for publication. *arXiv*, 1910.00279.

Hagmann, P., Cammoun, L., Gigandet, X., Meuli, R., Honey, C. J., Van Wedeen, J., and Sporns, O. (2008). Mapping the structural core of human cerebral cortex. *PLoS Biology*, 6(7):1479–1493.

Han, S., Mao, H., and Dally, W. J. (2015). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv:1510.00149*, pages 1–14.

Herculano-Houzel, S. (2009). The human brain in numbers: a linearly scaled-up primate brain. *Frontiers in human neuroscience*, 3:31.

Hochreiter, S. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(2):107–116.

Hong, S., Dissing-Olesen, L., and Stevens, B. (2016). New insights on the role of microglia in synaptic pruning in health and disease. *Current opinion in neurobiology*, 36:128–134.

Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment. Computing in Science and Engineering*, 9:90–95.

Jarrell, T. A., Wang, Y., Bloniarz, A. E., Brittin, C. A., Xu, M., Thomson, J. N., Albertson, D. G., Hall, D. H., and Emmons, S. W. (2012). The connectome of a decision-making neural network. *Science*, 337(6093):437–444.

J.G.White, E.Southgate, J.N.Thomson, S. (1986). The Structure of the Nervous System of the Nematode Caenorhabditis elegans Author ( s ): J . G . White , E . Southgate , J . N . Thomson , S . Brenner Source : Philosophical Transactions of the Royal Society of London . Series B , Biological Published by. *Philosophical Transactions of the Royal Society of London*, 314(1165):1–340.

Jonas, E. and Kording, K. P. (2017). Could a neuroscientist understand a microprocessor? *PLoS computational biology*, 13(1):e1005268.

Kerr, J. N., Greenberg, D., and Helmchen, F. (2005). Imaging input and output of neocortical networks in vivo. *Proc Natl Acad Sci USA*, 102(39):14063–14068.

Killian, N. J. and Bufalo, E. A. (2018). Grid cells map the visual world. *Nature neuroscience*, 21(2):161.

Kolb, B. and Gibb, R. (2011). Brain plasticity and behaviour in the developing brain. *Journal of the Canadian Academy of Child and Adolescent Psychiatry*, 20(4):265.

Krauss, P., Schuster, M., Dietrich, V., Schilling, A., Schulze, H., and Metzner, C. (2019). Weight statistics controls dynamics in recurrent neural networks. *PLoS ONE*, 14(4):1–13.

Kriegeskorte, N. and Douglas, P. K. (2018). Cognitive computational neuroscience. *Nature Neuroscience*, 21(9):1148–1160.

Lample, G. and Chaplot, D. S. (2017). Playing FPS games with deep reinforcement learning. *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, pages 2140–2146.

Latora, V. and Marchiori, M. (2001). Efficient behavior of small-world networks. *Physical review letters*, 87(19):198701.

Littman, M. (2012). Inducing Partially Observable Markov Decision Processes. *11th International Conference on Grammatical Inference*, 21:145–148.

Littman, M. L. (2001). Value-function reinforcement learning in Markov games. *Cognitive Systems Research*, 1:55–66.

Low, L. K. and Cheng, H.-J. (2006). Axon pruning: an essential step underlying the developmental plasticity of neuronal connections. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 361(1473):1531–1544.
Sparsity through evolutionary pruning prevents neuronal networks from overfitting

Lukoševičius, M., Jaeger, H., and Schrauwen, B. (2012). Reservoir Computing Trends. *KI - Künstliche Intelligenz*, 26(4):365–371.

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., and Moser, M.-B. (2006). Path integration and the neural basis of the 'cognitive map'. *Nature Reviews Neuroscience*, 7(8):663.

Mocanu, D. C., Mocanu, E., Stone, P., Nguyen, P. H., Gibescu, M., and Liotta, A. (2018). Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):2383.

Müller, M. and Wehner, R. (1988). Path integration in desert ants, cataglyphis fortis. *Proceedings of the National Academy of Sciences*, 85(14):5287–5290.

Müller, M. and Wehner, R. (1994). The hidden spiral: systematic search and path integration in desert ants, cataglyphis fortis. *Journal of Comparative Physiology A*, 175(5):525–530.

Nau, M., Schröder, T. N., Bellmund, J. L., and Doeller, C. F. (2018). Hexadirectional coding of visual space in human entorhinal cortex. *Nature neuroscience*, 21(2):188.

Oh, S. W., Harris, J. A., Ng, L., Winslow, B., Cain, N., Mihalas, S., Wang, Q., Lau, C., Kuan, L., Henry, A. M., Mortrud, M. T., Ouellette, B., Nguyen, T. N., Sorensen, S. A., Slaughterbeck, C. R., Wakeman, W., Li, Y., Feng, D., Ho, A., Nicholas, E., Hirokawa, K. E., Bohn, P., Joines, K. M., Peng, H., Hawrylycz, M. J., Phillips, J. W., Hofmann, J. G., Wohnoutka, P., Gerfen, C. R., Koch, C., Bernard, A., Dang, C., Jones, A. R., and Zeng, H. (2014). A mesoscale connectome of the mouse brain. *Nature*, 508(7495):207–214.

Oren-Suissa, M., Bayer, E. A., and Hobert, O. (2016). Sex-specific pruning of neuronal synapses in caenorhabditis elegans. *Nature*, 533(7602):206.

Paolicelli, R. C., Bolasco, G., Pagani, F., Maggi, L., Scianni, M., Panzanelli, P., Giustetto, M., Ferreira, T. A., Guiducci, E., Dumas, L., et al. (2011). Synaptic pruning by microglia is necessary for normal brain development. *science*, 333(6048):1456–1458.

Pascanu, R., Mikolov, T., and Bengio, Y. (2012). Understanding the exploding gradient problem. *arXiv:1211.5063v1*.

Perin, R., Berger, T. K., and Markram, H. (2011). A synaptic organizing principle for cortical neuronal groups. *Proceedings of the National Academy of Sciences*, 108(13):5419–5424.

Pierce-Shimomura, J. T., Morse, T. M., and Lockery, S. R. (1999). The fundamental role of pirouettes in Caenorhabditis elegans chemotaxis. *Journal of Neuroscience*, 19(21):9557–9569.

Ryu, W. S. and Samuel, A. D. T. (2002). Thermotaxis in Caenorhabditis elegans Analyzed by Measuring Responses to Defined Thermal Stimuli. *Journal of Neuroscience*, 22(13):1–7.

Schmidhuber, J. and Hochreiter, S. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Schrauwen, B., Verstraeten, D., and Van Campenhout, J. (2007). An overview of reservoir computing: theory, applications and implementations. *Proceedings of the 15th European Symposium on Artificial Neural Networks*, pages 471–82.

Séguinot, V., Cattet, J., and Benhamou, S. (1998). Path integration in dogs. *Animal behaviour*, 55(4):787–797.

Song, S., Sjöström, P. J., Reigl, M., Nelson, S., and Chklovskii, D. B. (2005). Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biology*, 3(3):0507–0519.

Sporns, O., Tononi, G., and Kötter, R. (2005). The human connectome: A structural description of the human brain. *PLoS Computational Biology*, 1(4):0245–0251.

Theves, S., Fernandez, G., and Doeller, C. F. (2019). The hippocampus encodes distances in multidimensional feature space. *Current Biology*, 29(7):1226–1231.

van den Heuvel, M. P. and Yeo, B. T. (2017). A Spotlight on Bridging Microscale and Macroscale Human Brain Architecture. *Neuron*, 93(6):1248–1251.

Van Der Walt, S., Colbert, S. C., and Varoquaux, G. (2011). The NumPy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 13:22–30.

Verstraeten, D., Schrauwen, B., D’Haene, M., and Stroobandt, D. (2007). An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403.

Watts, D. J. and Strogatz, S. H. (1998). Strogatz - small world network *Nature*. 393:440–442.

Wehner, R. and Wehner, S. (1986). Path integration in desert ants. approaching a long-standing puzzle in insect navigation. *Monitore Zoologico Italiano-Italian Journal of Zoology*, 20(3):309–331.

Wen, W., Wu, C., Wang, Y., Chen, Y., and Li, H. (2016). Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082.
Wierstra, D. and Wiering, M. (2004). Utile distinction hidden Markov models. *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004*, pages 855–862.

Xu, X., Ding, Y., Hu, S. X., Niemier, M., Cong, J., Hu, Y., and Shi, Y. (2018). Scaling for edge inference of deep neural networks. *Nature Electronics*, 1(4):216–222.

Yeo, W. and Gautier, J. (2004). Early neural cell death: dying to become neurons. *Developmental biology*, 274(2):233–244.
Supplementary Material

Figure S1: **Best network connections for all experiments and seeds.** The network connections of the best network visualised as a connected graph and a connection matrix for each seed (column) and each experiment (row).
Figure S2: **Validation fitness in the "control" experiment.** The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.
Figure S3: **Validation fitness in the "connection severance" experiment.** The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.
Figure S4: **Validation fitness in the "connection severance (low rate)" experiment.** The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.
Figure S5: Validation fitness in the "connection severance no mut" experiment. The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.
Figure S6: Validation fitness in the "connection severance sparsity reward" experiment. The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.
Figure S7: Validation fitness the "sparsity reward" experiment. The validation fitness over generation for different seeds (columns) and different validation mazes (rows). Shaded area denotes the range from minimum to maximum, dotted lines indicate the 25% and 75% percentile, and the solid line denotes the mean.