UNIVERSITY OF CALIFORNIA SAN DIEGO

Sleeping Networks: A Computational Model and Algorithm for the Role of Sleep in Learning and Memory

A dissertation submitted in partial satisfaction of the requirements for the degree Doctor of Philosophy

in

Neurosciences with a Specialization in Computational Neurosciences

by

Timothy Tadros

Committee in Charge:

Professor Maksim Bazhenov, Chair
Professor Marcus Benna
Professor Gert Cauwenberghs
Professor Eric Halgren
Professor Tatyana Sharpee

2022

The dissertation of Timothy Tadros is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

University of California San Diego

2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Throughout the writing of this dissertation, I have received a great deal of support and assistance. I would first like to thank my advisor, Professor Maxim Bazhenov for his support as the chair of my committee and my mentor. His expertise was invaluable and pushing me toward novel research directions. His willingness to teach as well as learn helped me make progress toward my dissertation and allowed me to contribute to multiple fields.

I would like to thank my committee for providing valuable feedback on my research progress and ensuring I'm headed in the right direction!

I would also like to acknowledge other collaborators in my laboratory, specifically Giri Krishnan and Oscar Gonzalez. Giri helped me hone my research ideas and questioned my ideas in ways that allowed me to sharpen my own thinking. Oscar was always willing to help whenever I had questions and made me feel comfortable from day one joining the lab.

In addition, I would like to thank my parents for their willingness to support my education and interest in my research. They have been there every step of the way on my path towards a PhD. Lastly, I would like to thank my friends who helped distract me from the struggles of a PhD and entertained me outside of research.

Chapter 1, in full, is a reprint of the material as it was submitted to The Journal of Neuroscience. Tadros, Timothy; Bazhenov, Maxim. Journal of Neuroscience, 2022. The dissertation author was one of the primary investigators and first author on this paper.

Chapter 2, in full, is a reprint of the material as it was published at the Bridging AI and Cognitive Science Workshop at ICLR 2020. Tadros, Timothy; Krishnan, Giri; Bazhenov, Maxim. BAICs Workshop, 2020. The dissertation author was one of the primary investigators and first author on this paper.

Chapter 3, in full, is a reprint of the material as it was published in the International Conference of Learning Representations 2020. Tadros, Timothy; Krishnan, Giri; Ramyaa, Ramyaa; Bazhenov, Maxim. ICLR, 2020 The dissertation author was one of the primary investigators and first author on this paper.

Chapter 4, in part, is a reprint of the material as it was submitted for publication. Tadros, Timothy; Krishnan, Giri; Ramyaa, Ramyaa; Bazhenov, Maxim. The dissertation author was one of the primary investigators and first author on this paper.

VITA

2017 Bachelor of Arts, Dartmouth College
2022 Doctor of Philosophy, University of California San Diego

PUBLICATIONS

Assessing Neural Network Scene Classification from Degraded Images. T. Tadros, N. Cullen, M. Greene and E.A. Cooper. ACM Transactions on Applied Perception 16.4 (2019):21.

Biologically Inspired Sleep Algorithm for increased generalization and adversarial robustness in deep neural networks. T. Tadros, G. Krishnan, R. Ramyaa, M. Bazhenov. ICLR 2020.

Role of Sleep in Formation of Relational Associative Memory. T. Tadros, M. Bazhenov. Journal of Neuroscience, in revision.

Sleep-like Unsupervised Replay Reduces Catastrophic Forgetting in Artificial Neural Networks. T. Tadros, G. Krishnan, R. Ramyaa, M. Bazhenov. Nature Communications, in revision.

Simulated sleep helps to generalize knowledge in a spiking network trained with spiketiming dependent plasticity. T. Tadros, G. Krishnan, M. Bazhenov. Bridging AI and Cognitive Science Workshop at ICLR, October 2019.

FIELDS OF STUDY

Major Field: Neuroscience

Studies in Memory Generalization and Computational Modelling of Sleep
Professor Maksim Bazhenov

Major Field: Computer Science

Studies in Artificial Neural Networks and Memory Generalization
Professor Emily Cooper

Major Field: Computer Science

Studies in Autonomous Vehicles and Route Planning
Professor Jonathan Sprinkle

ABSTRACT OF DISSERTATION

Sleeping Networks: A Computational Model and Algorithm for the Role of Sleep in Learning and Memory

by

Timothy Tadros

Doctor of Philosophy in Neurosciences with a Specialization in Computational Neurosciences

University of California San Diego 2022

Professor Maksim Bazhenov, Chair

This dissertation explores the computational role of sleep on memory consolidation. Chapter 1 explores a biophysical model of the thalamo-cortical network as it learns a relational memory task. We conclude by making predictions about the role of sleep on relational memory. Chapter 2 explores a less realistic biophysical model as it learns a digit recognition task. We created a sleep algorithm for this model and show that sleep can improve performance. Chapters 3 and 4 utilize this novel sleep algorithm for biophysical networks and apply this algorithm to artificial neural networks, who suffer from poor generalization (Chapter 3) and catastrophic forgetting (Chapter 4). We show that this sleep algorithm can help mitigate these issues and suggest that sleep is instrumental in memory generalization and continual learning.

1 Role of Sleep in Formation of Relational Associative Memory

1.1 Abstract

Relational memory, the ability to make and remember associations between objects, is an essential component of mammalian reasoning. In relational memory tasks, it has been shown that periods of offline processing, such as sleep, are critical to making indirect associations. To understand biophysical mechanisms behind the role of sleep in improving relational memory, we developed a model of the thalamocortical network to test how slow-wave sleep affects performance on an unordered relational memory task. First, the model was trained in the awake state on a paired associate inference task, in which the model learned to recall direct associations. After a period of subsequent slow-wave sleep, the model developed the ability to recall indirect associations. We found that replay, during sleep, of memory patterns learned in awake increased synaptic connectivity between neurons representing the item that was overlapping between tasks and neurons representing the unlinked items of the different tasks; this forms an attractor that enables indirect memory recall. Our study predicts that overlapping items between indirectly associated tasks are essential for relational memory, and sleep can reactivate pathways to and from overlapping items to the unlinked objects to strengthen these pathways and form new relational memories.

1.2 Introduction

The ability to form indirect associations between learned items with overlapping elements highlights an important part of abstract problem solving. This type of learning, known as transitive inference, is a fundamental feature of relational memory (DeVito, Kanter, and Eichenbaum 2010). For example, one may watch a movie (object A) and experience a feeling of familiarity about a certain actor (object B), giving rise to the question of what movie that actor has been in previously

(object C). This type of memory, where the premises that 'A goes with B', and 'B goes with C' are learned, represents a type of transitive inference where the indirect association (that 'A goes with C') is not inherently learned but is inferred by the subject. Despite the seeming complexity of the task, it has been shown that rats, primates, and humans are capable of performing transitive inference and relational memory tasks (Vasconcelos 2008; DeVito, Kanter, and Eichenbaum 2010). Importantly, depending on the type of task, the ability to connect indirect associations or inferences may not be explicitly acquired immediately after training (Ellenbogen et al. 2007; Walker et al. 2002).

Empirical studies suggest that offline processing, such as during sleep, is important in forming indirect associations (Ellenbogen et al. 2007; Werchan and Gómez 2013) . Sleep is a principle component behind many types of memory consolidation and plays an important role in learning (Klinzing, Niethard, and Born 2019; Walker and Stickgold 2004; Ji and Wilson 2007; Maquet 2001). The role of non-Rapid Eye Movement (NREM) sleep in learning and memory has been shown to be significant, aiding in consolidation of declarative memories and memories for complex motor learning tasks (Diekelmann and Born 2010; Walker et al. 2003; Miyamoto et al. 2016). A central hypothesis for memory improvement during NREM sleep is that replay or reactivation of learned synaptic memory traces during sleep oscillations (spindles or slow waves) strengthens synaptic traces of these labile memories (Wei, Krishnan, and Bazhenov 2016; Wei et al. 2018; González et al. 2020). Sleep has been shown to augment problem solving (Lewis, Knoblich, and Poe 2018a; Lau, Alger, and Fishbein 2011; Walker et al. 2002; Wagner et al. 2004; Nieuwenhuis et al. 2013) and hypothesized to create cognitive schemata by replaying memories with overlapping elements, strengthening the connections between overlapping memories and

leading to generalization of learned concepts (Lewis and Durrant 2011; Lewis, Knoblich, and Poe 2018a).

Accumulating evidence suggests that sleep may play a critical role in learning relational memory tasks (Ellenbogen et al. 2007; Werchan and Gómez 2013; Studte, Bridger, and Mecklinger 2015; Chatburn, Lushington, and Kohler 2014; Lau, Tucker, and Fishbein 2010; Lau, Alger, and Fishbein 2011)**.** One study showed that duration of slow wave sleep is significantly correlated with learning indirect associations (Lau, Tucker, and Fishbein 2010). Another study tested a subject's ability to relate abstract concepts through generalization, and found improvements after a day-time nap (Lau, Alger, and Fishbein 2011). It has also been shown that sleep can increase a subject's ability to perform hierarchical transitive inference, where A > B and B > C are learned premises and A > C is a tested abstraction (Ellenbogen et al. 2007).

Despite the progress made in understanding the role of sleep in increasing relational memory performance, it remains unknown what biophysical mechanisms account for this function. Here, using a biophysical model of the thalamocortical network, we tested the role of NREM sleep on the network's ability to perform a relational memory task. We found that the network can form indirect inferences, which were never trained directly, following periods of slow-wave sleep. We further revealed that sleep replay increases connections to/from a shared conjunctive memory unit, giving rise to an increase in performance during relational memory tasks. Ultimately, a theoretical understanding of how sleep aids with relational memory would guide development of experiments, where these findings can be tested *in vivo*.

1.3 Methods

1.3.1 Thalamocortical Network Model

1.3.1.2 Network Architecture

The base thalamocortical network used in this new study has been described in our other works (Wei et al. 2018; Wei, Krishnan, and Bazhenov 2016; Krishnan et al. 2016; González et al. 2020). The network was composed of two connected populations of neurons: thalamic and cortical. Different from previous work, we constructed two layers (functional regions) for both the thalamic and cortical components of the network and we did not rely on local connectivity but rather random connectivity between neurons. The thalamic part of the network was broken down into two populations and contained total 60 excitatory thalamocortical relay neurons (TC cells) and 60 inhibitory reticular neurons (RE cells). Layer 1 contained 40 TC neurons and 40 RE neurons, whereas layer 2 contained 20 TC and RE neurons. The cortical part of the network was also broken down into two layers, representing two functionally different cortical areas. In layer 1 (representing primary visual cortex), there were 200 excitatory pyramidal neurons (PY cells) and 40 inhibitory interneurons (IN cells). In layer 2 (representing associative cortex) there were 100 PY neurons and 20 IN cells. Connectivity was random; excitatory connections were mediated by NMDA and AMPA connections, while inhibitory connections were mediated by $GABA_A$ and $GABA_B$ connections. All connections are summarized in Table 1.1 and described as follows.

To describe specific connections, starting in the thalamus, RE neurons received AMPA connections from TC neurons and $GABA_A$ connections from RE neurons as well as AMPA connections from PY neurons in associated cortical layer. AMPA synapses between TC and RE cells had connection probability 10% in layer 1 and 20% in layer 2. RE cells were connected to each other through $GABA_A$ synapses within the same layer with probability 6.25% in layer 1 and

12.5% in layer 2. Finally, cortical PY neurons synapsed via AMPA connections onto RE cells with connection probability 10% and 20% in layers 1 and layer 2, respectively. TE cells received connections from RE cells through both $GABA_A$ and $GABA_B$ synapses, as well as AMPA connections from PY neurons in associated cortical layer. Each TC cell received a connection from an RE cell with a 10%, and 20% probability in layer 1 and layer 2, respectively. Each TC cell also received an AMPA synapse from cortical PY neurons, with connection probability 12.5% and 25%, in layers 1 and layer 2, respectively.

Table 1.1: Thalamocortical network connection architecture

Connection Type	Synapse Type	Connection Prob. (L1, L2)
TC – RE	AMPA	10%, 20%
RE – TC	$GABA_A$, $GABA_B$	10%, 20%
RE – RE	$GABA_A$	6.25%, 12.5%
PY – RE	AMPA	10%, 20%
PY – TC	AMPA	12.5%, 25%
TC – PY	AMPA	10%, 20%
PY(L1) – PY(L2)	AMPA (plastic), NMDA	20%
PY(L2) – PY(L2)	AMPA (plastic), NMDA	50%
PY(L2) – PY(L1)	AMPA	25%
IN – PY(L1)	$GABA_A$	2 connections
IN – PY(L2)	$GABA_A$	13 connections
PY(L1) – IN	NMDA, AMPA	3.75%, 7.5%
PY(L2) – IN	NMDA, AMPA	5%, 10%
TC - IN	AMPA	5%, 50%

In the cortex, PY neurons received non-plastic AMPA connections from TC cells, plastic and non-plastic AMPA connections from other PY neurons, and $GABA_A$ connections from IN neurons in the same cortical layer. The TC cells in layer 1 connected to PY neurons in layer 1 of cortex with a connection probability of 10%. In layer 2, this connection probability was increased to 20%. Thus, considering size difference between layer 1 and 2, each PY neuron received about the same number of TC inputs. In layer 1 of cortex, each PY neuron received feedback connections

from layer 2 PY neurons with a connection probability of 25%. In addition, each layer 1 PY neuron received two inhibitory $GABA_A$ connections from IN neurons in layer 1 of cortex. In layer 2, each neuron received a feedforward plastic AMPA connection from a layer 1 PY neurons with probability 20%, and a recurrent plastic AMPA connection from layer 2 PY neurons with probability 50%. Each plastic AMPA connection in cortex was also accompanied by a non-plastic NMDA excitatory synapse. In addition, layer 2 PY neurons received 13 $GABA_A$ connections from local INs.

Finally, each IN received non-plastic AMPA connections from TC cells in thalamus, with connection probability 3.75% in layer 1 and 7.5% in layer 2. In addition, all INs received non-plastic NMDA and AMPA synapses from PY neurons in both layer 1 and layer 2 of cortex. Layer 1 PY to layer 1 and layer 2 INs AMPA and NMDA connections occurred with a probability of 5% and 10%, respectively. Layer 2 PY to layer 1 and layer 2 INs AMPA and NMDA connections occurred with a probability of 5% and 50%, respectively. Note that the latter 50% connections were much weaker than other connections.

1.3.1.2 Wake-sleep Transition

The transition between wake and sleep was modelled after previous work which describes the role of neuromodulators - acetylcholine (ACh), histamine (HA), and GABA - during the sleep and waking state needed to observe sleep rhythms canonical of slow-wave sleep (Krishnan et al. 2016). ACh modulated potassium leak currents in all neuron types and excitatory AMPA connections within cortex. HA modulated the strength of the hyperpolarization-activated mixed cation current in TC neurons and GABA modulated the strength of inhibitory GABAergic synapses in both thalamus and cortex. The levels of ACh and HA were reduced during stage 3 (N3) slow-wave sleep (SWS) while GABA levels were increased compare to awake state. The

exact levels of each neuromodulator were chosen by conducting a parameter sweep and observing which parameters resulted in the appearance of canonical slow waves. In addition, to simulate stage 2 (N2) sleep characterized by spindles, neuromodulation parameters were determined by parameter sweep looking for the local field potential (LFP) power in the spindle frequency band (7-16 Hz in our study). Parameters for N2 sleep were intermediate between waking and N3 states.

1.3.1.3 Intrinsic Currents

All neurons were modelled with Hodgkin-Huxley kinetics and equations can be found in previous works (Wei et al. 2018; González et al. 2020). In cortex, PY and IN neurons possessed dendritic and axo-somatic compartments (Wei et al. 2018). Membrane potential dynamics were modeled by the following dynamical equations:

$$C_m \frac{dV_D}{dt} = -I_D^{Na} - I_D^{NaP} - I_D^{Km} - I_D^{KCa} - ACh_{gkl} I_D^{KL} - I_D^{HVA} - I_D^{L} - g(V_D - V_S) - I^{syn},$$

$$g(V_D - V_S) = -I_S^{Na} - I_S^{NaP} - I_S^{K},$$

where C_m is membrane capacitance, V_D and V_S are the dendritic or axo-somatic membrane voltages, respectively, I^{Na} is the fast sodium (Na^+) current, I^{NaP} is the persistent sodium current, I^{Km} is the slow voltage-dependent non-inactivating potassium (K^+) current, I^{KCa} is the slow calcium (Ca^{2+})-dependent K^+ current, ACh_{gkl} is the change in K^+ leak current I^{KL} which depends on the level of acetylcholine (ACh) which changes during wake and sleep states, I^{HVA} is the high-threshold Ca^{2+} current, I^L is the chloride (Cl^-) leak current, g is the conductance between dendritic and axo-somatic compartments, and I^{syn} is the synaptic current input to the neuron (as described in the next section). INs contained all the above intrinsic currents with except of the persistent sodium current. All intrinsic ionic currents (I^j) were modeled based on Hodgkin-Huxley (Hodgkin and Huxley 1952) equations as follows:

$$I^j = g_j m^M h^N (V - E_j),$$

where g_j is the maximal conductance, m (activation) and h (inactivation) are the gating variables, V is the voltage of the compartment, and E_j is the reversal potential of the current. Gating variable dynamics were described as follows:

$$\frac{dx}{dt} = -\frac{x - x_\infty}{\tau_x},$$

$$\tau_x = \frac{(1/(\alpha_x + \beta_x))}{Q_T},$$

$$x_\infty = \frac{\alpha_x}{(\alpha_x + \beta_x)},$$

where $= m$ or h, τ is the time constant, Q_T is the temperature related term, $Q_T = Q^{((T-23)/10)} = 2.9529$, with $Q = 2.3$ and $T = 36$.

In the thalamus region of the model, TC and RE neurons were modeled by single compartment neurons with the following dynamical equation:

$$C_m \frac{dV_D}{dt} = -I^{Na} - I^K - ACh_{gkl} I^{KL} - I^T - I^h - I^L - I^{syn},$$

where I^{Na} is the fast Na^+ current, I^K is the fast K^+ current, I^{KL} is the K^+ leak current, I^T is the low-threshold Ca^{2+} current, I^h is the hyperpolarization-activated mixed cation current, I^L is the Cl^- leak current, and I^{syn} is the total synaptic current input to the neurons (described in next section). The hyperpolarization-activated mixed cation current I^h was not expressed in RE neurons. In addition, histamine (HA) exerted its influence on I^h in TC cells by shifting the activation curve of HA_{gh} as described by:

$$m_\infty = \frac{1}{1 + exp\left(\frac{V + 75 + HA_{gh}}{5.5}\right)}.$$

Our previous work gives a more detailed description of the individual currents (Krishnan et al. 2016; Wei et al. 2018).

1.3.1.4 Synaptic Currents and Spike-Timing Dependent Plasticity (STDP)

Here, we describe the synaptic currents which were composed of AMPA, NMDA, $GABA_A$, and $GABA_B$ synapses as well as the STDP rules (see Krishnan et al., 2016; Wei et al., 2018 for more details on the specific synaptic currents). The effect of acetylcholine on AMPA and $GABA_A$ synaptic currents was described by the following equations:

$$I_{syn}^{AMPA} = ACh_{AMPA}\, g_{syn}\, [O](V - E_{syn}),$$

$$I_{syn}^{GABA} = \gamma_{GABA_A}\, g_{syn}\, [O](V - E_{syn}),$$

where g_{syn} is the maximal conductance, $[O]$ is the fraction of open channels, and E_{syn} is the reversal potential of the channel ($E_{GABA\text{-}A}$ = -70 mV, E_{AMPA} = 0 mV, and E_{NMDA} = 0 mv). ACh_{AMPA} describes the influence of acetylcholine levels on AMPA synaptic currents for PY-PY, TC-PY, and TC-IN connections. γ_{GABA_A} modulated the GABA synaptic currents for inhibitory IN-PY, RE-RE, and RE-TC connections. These values were changed between sleep and wake states. The influence of GABA was increased during sleep so that γ_{GABA_A} was increased, whereas ACh was decreased during sleep so that ACh_{AMPA} was reduced. During stage 3 (N3) sleep, the model generated periodic transitions between Up and Down states. As in our previous studies, Down-to-Up transitions were mediated by spontaneous miniature excitatory transmitter release from PY-

PY and PY-IN synapses, while Up-to-Down transitions depended on synaptic depression and intrinsic current, such as $I_{K(Ca)}$ (I. Timofeev et al. 2000).

Spike-timing dependent plasticity (STDP) controlled long-term potentiation and depression of synaptic weights between PY neurons. The change in the synaptic strength (g_{AMPA}) and amplitude of miniature EPSPs (A_{mEPSP}) were described previously (Wei et al. 2018):

$$g_{AMPA} \leftarrow g_{AMPA} + g_{max}\, F(\Delta t),$$

$$A_{mEPSP} \leftarrow A_{mEPSP} + f A_{PY-PY}\, F(\Delta t),$$

where g_{max} is the maximal conductance of g_{AMPA}, and f=0.01 represents the lower effect of STDP on A_{mEPSP} as compared to g_{AMPA}; F represents the STDP function and depends on the relative timing of pre- and post-synaptic spikes as defined by:

$$F(\Delta t) = \begin{cases} A_+\, e^{-|\Delta t|/\tau_+}, if\ \Delta t > 0 \\ -A_-\, e^{-|\Delta t|/\tau_-}, if\ \Delta t < 0 \end{cases}$$

where $A_{+,-}$ set the maximum amplitude of synaptic change ($A_{+,-} = 0.002$, $\tau_{+,-} = 20$ms). A_-, the synaptic depotentiation term, was reduced to 0.001 during training to reflect the effect of acetylcholine during focused learning (Sugisaki et al. 2016).

1.3.1.5 Heterosynaptic Plasticity

Heterosynaptic plasticity was implemented in some simulations. To mimic heterosynaptic plasticity properties observed *in vivo* (Chistiakova et al. 2014; Volgushev et al. 2016), after each STDP event in which a synaptic weight was modified, we also modified the weights of remaining synapses into the same neuron to hold the total synaptic input per neuron constant. Thus, if $s_i = \sum_{j=1}^{n} w_{ij}$ is the total synaptic input to neuron *i* from neurons *j = 1:n*, then this quantity was maintained constant throughout the simulation. Thus, any increase of a single synaptic weight would result in a corresponding decrease of the other weights connecting to the same neuron *i*. To

implement this property, we computed the total synaptic input for each neuron *i* after supervised training was completed. Then, during associative training, after each STDP event, the new conductances for all pre-synaptic neurons *j* were computed by setting $w_{ij} = w_{ij} * s_i^{supervisedTR} / s_i^t$, where $s_i^{supervisedTR}$ is the synaptic input to neuron *i* after supervised training and s_i^t is the current total synaptic input to neuron *i* at time *t* of the STDP event.

1.3.1.6 Memory Training and Testing

Training and testing of associative memories was modelled after behavioral works (Lau, Tucker, and Fishbein 2010). After creating a two-layer cortical architecture, we selected the groups of neurons in each layer that correspond to each stimulus. Neuron IDs were mapped to a stimulus label as shown in Table 1.2. The first training phase was the supervised learning. Here, an individual item was stimulated in layer 1 followed, with 5 ms delay, stimulation of that item in layer 2. This phase created a feedforward pathway through the network that represents an individual stimulus. Each feedforward pathway stimulation (e.g., A-A') included 40 trials with a 500ms gap between trials. The total length of supervised training was therefore 120 seconds for all 6 feedforward pathways.

Following supervised training, we implemented an unsupervised associative training phase, where pairs of stimuli were presented simultaneously. This occurred by stimulating pairs of input items together (e.g., A+B, B+C, etc) in layer 1. These pairs of items were stimulated sequentially every 500 ms with a 2 second gap between same-pair stimulations. The exact duration of associative training varied by experiment, but if associative training time was 135s/pair, then each pair was stimulated 270 times.

Finally, there was a sleep phase. During sleep, the levels of neuromodulators were changed to induce spindles (N2) or slow oscillations (N3) and there was no external stimulation provided.

Each sleep phase was followed by a testing phase, where each of the six groups was stimulated in layer 1, and the response of layer 2 neurons was measured. Stimulation was provided every 500 ms and each group was stimulated eight times. Performance was measured as the network's ability to recall both the direct and indirect associated item (e.g., upon stimulation of A, can the network recall both B' and C'?). In Figure 1.9B, we performed additional tests where groups A, C, X, and Z were stimulated and neuron groups B' and Y' were hyperpolarized to prevent activation. In another experiments, we hyperpolarized neurons from linking groups B/B' and Y/Y' during sleep to simulate experiments with optogenetic inactivation.

Table 1.2: Neuron indices in cortical architecture

Neuron Groups	Layer One Region	Layer Two Region
A,A'	Neurons 10-29	10-19 (Neurons 210-219)
B,B'	Neurons 40-59	20-29 (Neurons 220-229)
C,C'	Neurons 70-89	30-39 (Neurons 230-239)
X,X'	Neurons 110-129	50-59 (Neurons 250-259)
Y,Y'	Neurons 140-159	60-69 (Neurons 260-269)
Z,Z'	Neurons 170-189	70-79 (Neurons 270-279)

1.3.2 Experimental Design and Statistical Analysis

All analyses were performed within standard Python functions and libraries. Data are presented as mean and standard deviation unless otherwise stated. Each experiment was repeated with 10 network stimulations from different network initializations and random seeds for purposes of statistical analyses, using standard two-sided or one-sided T-tests.

1.3.2.1 Relational Memory Performance Metrics

Here, we describe the association matrices shown in Figure 3 as well as the conversion from these matrices to an association score. To build an association matrix, individual neuronal groups were stimulated in layer 1 (e.g., item A was stimulated), and we measured the number of spikes in each of the six layer 2 groups (A', B', C', X', Y', Z'). This number was averaged over

the 10 different (initialization) network simulations and 8 testing trials within each network simulation. We only considered spikes that occur within 150ms of stimulation to the layer 1 groups. To compute an association score based on the association matrix, we built a binary 6x6 mask with 1's in the upper left and lower right 3x3 grids and -1's everywhere else. This mask depicts what an ideal associative matrix should look like, where activity in the upper left and lower right grids is acceptable and activity in the upper right and lower left grids is spurious. After element-wise multiplication of the mask and the associative matrix, the resultant matrix was summed up across both rows and columns. To normalize this final score, we divided the final sum by the maximum element in the association matrix multiplied by 18 (here, 18 is the number of elements that should be positive, e.g. number of groups * number of items in each group, or 6*3, where 6 is number of groups (A-B-C, X-Y-Z) and 3 is number of items in each group). The final number was on a scale from -1 to 1, where a score of -1 occurs when the association matrix is the opposite of what it should be after successful learning (e.g., stimulating group A activates X', Y', and Z'), an association score of 0 is true for a random matrix, and an association score of 1 indicates perfect performance on the task (e.g., stimulating group A equally stimulates A', B', C').

1.3.2.2 Latency and Rate Analysis

In Figure 1.4, we show the spiking rates and latency of neurons in layer 2. To compute the latency of response, after applying a pulse of stimulation during testing, we analyzed the next 200 ms window of activity in layer 2. The latency, for each layer 2 neuron, was determined by taking the time of activation of a neuron in layer 2 and subtracting the time of stimulation in layer 1. If a neuron does not spike in the 200ms time window, its latency was ignored from the computation. The firing rate was computed by calculating the total number of spikes that occur in the 200 ms window. We considered four different types of memories: direct memories (e.g., activation of

neuron group A'/C' when B' is stimulated), indirect memories (e.g., activation of neuron group A'/C' when C/A is stimulated, respectively) and incorrect memories (e.g., activation of neuron group X'/Z' when A is stimulated). For each type of memory, latencies and rates were averaged across all pairs of that type (e.g., direct memories = A-B', B-A', B-C', indirect memories = A-C', C-A', incorrect memories = A-X', A-Z', C-X', C-Z' for the ABC triplet). We should note that this metric likely overestimated latency for the incorrect memories, since it did not consider the fact that if a neuron does not fire, its latency is ignored from the computation. Thus, e.g., if only one incorrect neuron fired with a latency of <50 ms, then the average latency would in fact be <50 ms. This was rarely the case but nevertheless the drop in latency of the incorrect memories was likely due to this phenomenon since the rate of firing (3 spikes/stimulation) is quite low already.

1.3.2.3 Weight Analysis

In Figure 1.5 and 1.6A-D, we explored the synaptic connectivity matrices. Figure 1.5 was obtained by recording the synaptic weights between neurons for each type of connection (feedforward or recurrent). To evaluate the synaptic input to each neuron *i*, we computed the following equation: $s_i = \sum w_{\{ij\}}$, where *j* is any neuron that meets the criteria (e.g., direct, indirect, or incorrect memories) and $w_{\{ij\}}$ is the weight connecting form neuron *j* to neuron *i*. If a synapse does not exist between two neurons *i* and *j*, then the weight is ignored. In Figures 6A-D, we built a graph of all neurons in layer 2. A node in the graph depicts 10 individual neurons from layer 2. An edge was created between nodes, if there existed a weight that exceeds 80% of the maximum weight value at that given time point. For example, if the maximum weight at time *t* is W_{max}^{t} , then the threshold is defined as $W_{th} = 0.8 * W_{max}^{t}$. For any weight, in the weight matrix, an edge was created between two nodes if there existed a weight value that exceeds the value W_{th} and the thickness of the edge depicts how many such weights meet the criteria.

1.3.2.4 Modularity Analysis

Community detection algorithm was used to describe brain network changes during task learning (Bassett et al. 2015; Alexander-Bloch et al. 2010; Mucha et al. 2010). Modularity refers to the formation of cliques in a network, or series of intra-connected nodes with limited connections to other cliques (Alexander-Bloch et al. 2010). Time-dependent communities can be analyzed by measuring the structure of multi-slice networks, which can be thought of as a combination of individual networks that are composed of nodes that are linked in time to past and future versions of that network (Mucha et al. 2010). To perform community detection (Figures 6E-F), we used existing community detection algorithm (Jeub et al. 2020). First, the Leicht-Newman modularity matrix for ordered and directed layers was computed (Leicht and Newman 2008). This algorithm finds a partition that maximizes the modularity of the matrix. After this partition was computed, the generalized Louvain method for community detection was applied (Jeub et al. 2020; De Meo et al. 2011). As a result of applying these algorithms, a network partition and community assignment graph was returned as a function of time. The algorithms aim to find a community assignment partition that maximizes the resulting modularity of the network. Two parameters were tuned to aid in this process: the coupling between temporal layers ($\omega = 1.0$) and the intra-layer resolution ($\gamma = 1.75$).

1.3.2.5 Replay Analysis

To analyze memory replay, we adopted a method from (González et al. 2020). First, the LFP during sleep was computed by evaluating the average membrane potential across all pyramidal neurons in the cortex. A threshold for crossing from Up to Down state and vice versa of the slow oscillations was computed by taking the resting membrane potential (-63 mV) and subtracting the mean sleep membrane potential. After the threshold was computed, we filtered the

LFP using a 2nd-order Butterworth filter with a Nyquist frequency of 500 Hz and passband and stopband frequencies of 0 and 3 Hz, respectively. Next, we applied the threshold to find the Up to Down state and Down to Up state transition times. Activity above the threshold was denoted as an Up state.

Once the Up and Down states were identified, we analyzed the activity within each individual Up state to calculate replay events. A spiking event was considered a replay event when a pre-synaptic and a post-synaptic neuron fired within a given time window (<200 ms). The order of firing (pre-post, or post-pre) was used to determine the direction of replay and to compute a directional graph between neurons, where each edge stores the number of replay events going in that direction (see (González et al. 2020) for details).

1.4 Results

1.4.1 Thalamocortical Model of Relational Memory

In this work, we used a minimal thalamocortical network model to test the role of sleep in learning an unordered relational memory task (Figure 1.1A-B). Cortex was modelled with a network consisting of two layers, each representing a distinct functional area of the cortex, and each including excitatory pyramidal (PY) cells and inhibitory interneurons (INs). A two-layer cortical model was motivated by visual associative learning in the primate brain. Prior work suggests that associations are learned by recurrent synaptic connections in the parietal associative cortex (Bjekić et al. 2019; Fitzgerald, Freedman, and Assad 2011; Fitzgerald et al. 2013; Aminoff and Tarr 2015). This area of cortex receives input from primary visual cortex (Galetti et al., 2001 Eur. J. Neuro.), which shows a mostly stereotyped response upon presentation of visual stimuli (Deitch et al., 2021). Thus, we constructed our model with two populations of cortical neurons (which we call layers here, when we refer to the model): the first representing visual cortex with a

mostly stereotyped population response to specific stimuli, and the second representing associative cortex, with recurrent connectivity to promote associative memory learning.

Thalamus was modelled by two populations of neurons, each including excitatory thalamocortical (TC) neurons and inhibitory reticular (RE) neurons, with bidirectional connections to its respective cortical areas (see Methods for details). Indeed, neuroanatomical studies suggest that different subdivisions of thalamus project to different areas of cortex, with primary areas of thalamus such as LGN projecting bi-directionally to primary visual cortex (Briggs et al., 2007), and other subdivisions, such as the lateral posterior nucleus, connecting bi-directionally to parietal cortex (Lyamzin et al., 2019 Neurosci. Research). All neurons were simulated with Hodgkin-Huxley dynamics and are based upon previous work (Wei et al. 2018; Wei, Krishnan, and Bazhenov 2016; Krishnan et al. 2016).

Using this model, we were able to simulate three distinct states of the network – awake, stage 2 (N2) sleep and stage 3 (N3) sleep – by changing the level of neuromodulators (Krishnan et al. 2016; Vanini, Lydic, and Baghdoyan 2012). Awake state was characterized by random asynchronous firing of cortical neurons, N2 sleep was characterized by spindles with occasional Down states, and N3 sleep (or slow-wave sleep (SWS)) was characterized by canonical slow oscillations between Up (active) and Down (silent) states (Blake and Gerard 1937; Steriade 2006; Steriade, McCormick, and Sejnowski 1993) (Figure 1.1B, C, see also Figure 1.8A). The thalamic component of the network primarily served the function of driving and modulating oscillations during sleep, specifically to increase synchrony of sleep slow oscillations in N3 (Lemieux et al. 2014) and to generate spindles in N2, while learning-related plasticity occurred in the cortical neuronal populations. Synaptic plasticity was implemented in AMPA receptors, occurring in feedforward connections between layer 1 and layer 2 cortical pyramidal cell populations, as well

as recurrent connections between layer 2 pyramidal neurons (Figure 1.1F) (see Methods for details).

To test relational memory in the model, we built two triplets of relational memory items (ABC, XYZ). During associative training, each of the four direct object pairs (A-B, B-C, X-Y, Y-Z) was presented to the network, as described below (Figure 1.1A, left). During testing, a single item from each pair was presented (e.g., item A) and the ability of the network to recall each of the relevant associative items (items B and C) was measured (Figure 1.1A, right). Each of the six distinct items (A, B, C, X, Y, Z, Figure 1.1F) were represented by distinct groups of neurons in the first layer of the network.

1.4.2 Training and Testing Stimulation Protocol

The network stimulation included three distinct phases: supervising training, associative training, and sleep (Figure 1.2A). The first phase in training was to build connections between neurons representing item A in the first layer (neuron group A) and "higher level" neurons representing the item in the second layer (neuron group A') (Figure 1.2B). Since all connections in the model were initially random, before training there were equal connections from neuron group A to all the neuron groups in the second layer (A'-Z'). Thus, to create distinct pathways through the cortex that represent each of the six distinct items, we incorporated the supervised training phase. During supervised training, neuron group A was stimulated and then neuron group A' in the second layer was stimulated with a 5 ms time delay. Through spike-timing dependent plasticity (STDP), this stimulation paradigm strengthened feedforward connections between A and A' and led to the formation of a pathway through the network representing each of the six distinct items. After supervised training, there was a testing phase where each of the six neuron groups in

layer 1 was stimulated and the activity of neurons in associative layer 2 was measured. During testing, plasticity was turned off so spiking activity did not lead to STDP events.

Following supervised training, we simulated associative learning phase. Items A+B, B+C, X+Y, Y+Z were presented simultaneously to the network by stimulating groups A and B together or B and C together, etc. (Figure 1.2C). Because of the preceding supervised training, neurons in the second layer responded to the stimulation in the first layer, such that, e.g., when neuron groups A and B were stimulated, neuron groups A' and B' fired without any direct stimulation. After a period of associative training, there was another testing phase.

During the associative training phase, we also tested two plasticity schemes: In the first scheme, STDP was used as a sole learning rule to increase synaptic connectivity between neurons with correlated firing activity and decrease synaptic connectivity between those neurons with uncorrelated firing activity. In the second scheme, STDP was used along with hetero-synaptic plasticity (Chen et al. 2013; Chistiakova et al. 2014). Heterosynaptic plasticity can induce plastic changes at synapses that are not active during the induction. It has been postulated since early theoretical studies which used normalization to prevent runaway dynamics of synaptic weights and introduce synaptic competition to the model systems with Hebbian-type learning (von der Malsburg 1973; Miller 1996). Any synapse to a cell may express heterosynaptic changes after episodes of strong postsynaptic activity leading to a sufficient rise of intracellular calcium (reviewed in (Chistiakova et al. 2014; 2015)). Thus in the model including heterosynaptic plasticity, after each STDP event, individual weights connecting to a neuron were modified so that the total sum of synaptic inputs to the neuron remained constant. This served to balance excitation in the network and prevent runaway networks dynamics by ensuring that the overall level of

excitation remains constant during learning. Below we report results for each of these conditions and we discuss later possible implications of heterosynaptic plasticity in associative learning.

Finally, we simulated sleep phase (Figure 1.2D). Based on experimental data, the improvement of indirect relational memory following sleep is most correlated with slow-wave sleep (SWS) (Tucker, Fishbein, and Lau 2010) and thus we primarily focused on testing the effect of SWS on relational memory (differential role of spindles is discussed later in the paper). We need to mention that we did not explicitly model hippocampus and associated ripple events; instead, we assumed that coactivation of the cortical neurons (e.g., A+B) may be result of direct sensory input or hippocampal input (as postulated by 'indexing' theory (Teyler and DiScenna 1986)). Following SWS, there was another testing phase. Overall, based on behavioral work, we tested the hypothesis that following sleep, the presentation of item A in the first layer will lead to a greater co-activation in neuron groups A' and C', i.e., association between items A and C would form, when compared with the same group activation before sleep.

1.4.3 Sleep Improves Associative Memory Performance Both With and Without Heterosynaptic Plasticity

In Figure 1.3A, the strength of response in the layer 2 neuronal subgroups (A'-Z') is shown in response to stimulation of each of the six layer 1 neuronal subgroups (A-Z) in the first cortical layer. After supervised training, stimulation of a single group in layer 1 (e.g. group A) led to activity in its corresponding neuronal subgroup in layer 2 (group A'). Spurious activity in other layer 2 groups was usually minimal and based off the random connectivity matrix, where some groups may be connected (based on number of connections) more strongly than other groups (Figure 1.3A, left), (See Methods and materials for computing activity).

After associative training, an increase in direct relational memory was observed. Here, stimulation of a neuron group A led to activity in neuron groups A' and B', indicating that the

network has learned to make direct associations between objects A and B. Stimulation of the linking item (e.g. B or Y) led to activity in all three of the items in the corresponding triplet (A', B', C' or X', Y', Z'). However, most notable is that stimulating A or C alone did not lead to a strong response in the indirect relational item, C' or A', respectively (Figure 1.3A, middle). After sleep phase, this indirect relational memory was significantly strengthened, as stimulation of A or C (X or Z), led to a stronger response in the indirect relational item, C' or A' (Z' or X'), respectively (Figure 1.3A, right).

To quantify the changes in the association matrices, we used a measure of how "diagonal" the matrix is in respect to four main 3x3 blocks, which evaluated the extent to which the matrix shows strong responses in the upper left and lower right 3x3 blocks, and low responses in the top right and bottom left 3x3 blocks (see Methods and Materials). (This measure would be zero for uniform matrix; +1 for a matrix with the top left and bottom right 3x3 blocks all having the same values, with zero activity in the top right and bottom left 3x3 blocks; and -1 for the opposite case (activity in top right and bottom left blocks)). We found that sleep leads to a significant improvement in relational memory, based on simulating ten random different network configurations (Figure 1.3B, $p=0.0062$, $t(9)=3.55$, between relational memory after sleep and after associative training, based on a two-sided t-test).

The extent of improvement after sleep was determined by two factors: the length of associative training and length of sleep. We observed that if associative training was long, then indirect associations can be learned without sleep (Figure 1.3E, 50 seconds). However, when associative training was shorter, then sleep had a beneficial impact on improving relational memory (Figure 1.3E, 20, 35 seconds). Given the model with no homeostatic mechanisms built in to constrain synaptic weights, it was observed that long training or long sleep periods could lead

to runaway network dynamics, where stimulating a single neuronal group in layer one leads to activity across many neurons of the second layer, thus lowering overall response specificity and performance.

Given the negative impact of the runaway network dynamics, we next explored the use of biologically realistic heterosynaptic plasticity mechanism to constrain synaptic weights. Thus, during associative training, heterosynaptic plasticity was put in place, such that the total sum of synaptic inputs to any neuron was conserved over time. In this model, any event that leads to synaptic potentiation between neurons would also lead to a corresponding depotentiation of other connections to the same neuron to keep net sum of all input weights constant (see Methods and Materials for details). In the model with heterosynaptic plasticity, we observed less spurious activity after associative training (Figure 1.3C, middle). In addition, activation of the indirect memory after associative training was almost non-existent. Importantly, after sleep, the activity in the indirect memory items was strong, with very little activity in neurons representing non-associated items (Figure 1.3C, right). Here, improvement after sleep was strongly significant (Figure 1.3D, $p=3.78 \times 10^{-6}$, $t(9) = 13.04$, based on two-sided T-test). This suggests that, for SWS to have a beneficial impact on the network's ability to recall indirectly associated items, the weights before sleep must be sufficiently separated but not too strong overall, as it was when heterosynaptic plasticity was applied during associative training. In general, the best performance was observed when sleep was incorporated into the network (Figure 1.3F). Increasing the training time beyond a certain duration did not always increase the baseline performance; however, sleep applied even after long associative training could still further improve performance. We tested how associative memory performance depends on the total number of slow waves and we found a significant positive correlation in a broad range of sleep durations (Fig. 3G). This result is in

agreement with previous experimental work that found a significant correlation between the SWS length and relational memory learning (Tucker, Fishbein, and Lau 2010). Interestingly, very long sleep could have the opposite effect and reduce performance (see, e.g., T_{sleep}=700sec), suggesting the existence of an optimal sleep duration that could also depend on the duration of preceding training sessions. For further analyses, we used the heterosynaptic plasticity condition with T_{sleep} = 300 seconds and T_{train} = 135 seconds.

Synaptic plasticity may also occur between cortical pyramidal cells and interneurons, as well as between thalamus and neocortex. Although we did not explicitly incorporate these types of plasticity in our model, we tested effect of changes in the balance of excitation and inhibition on post-sleep memory performance. Thus, we modified the level of inhibition in the network by setting it to +/- 10% of the baseline value. We found no significant difference in the associative score after sleep ($t(10) = -0.8$, $p = 0.4$, one-sided t-test). After associative training performance was relatively higher in the network with reduced baseline inhibition ($t(10) = 2.4$, $p= 0.02$, one-sided t-test). In this case, there was still a significant post-sleep improvement ($t(10) = -4.96$, $p = 0.0001$). The network with increased inhibition revealed slightly reduced performance right after associative training but relatively higher gain after sleep.

1.4.4 Sleep Increases Amplitude and Decreases Latency of Indirect Memory Response

Since sleep increases the association score, we next asked if sleep can improve the latency of group activation by reducing time delay between responses of stimulated and indirectly recalled groups. To test this, we analyzed the raw neuronal traces after supervising training, after associative training, and after sleep (Figure 1.4A-C). As mentioned before, heterosynaptic plasticity was in place in all these simulations. During testing, each group (A-C, X-Z) was simulated eight times every 500 ms in layer 1 and the response in layer 2 was measured. We next

converted these firing patterns into a local field potential (LFP) for each of the six groups of neurons in the second layer and averaged across eight simulations. Results are shown when X is stimulated in the first layer (Figure 1.4D). After supervised training, stimulating X led to a strong response in X' (Figure 1.4D, left). After associative training, the strength of the response of Y' was increased and there was a small, sustained response in Z' (Figure 1.4D, middle). Finally, after sleep the response profiles of Y' and Z' nearly become overlapping, suggesting that the network has used its knowledge of an association between Z' and Y' to correctly infer the indirect association between Z' and X' (Figure 1.4D, right).

We measured response latency as a time delay from layer 1 stimulation to the first action potential in each layer 2 neuronal group's response and we measured response intensity as total number of spikes per stimulation of each layer 2 neuronal group. After supervised training, the average latency of direct memories (A-B', which have not been learned yet), indirect memories (A-C'), and incorrect memories (A-X') were all similar at ~200ms (Figure 1.4E, left group). In addition, the rate of response was very low and similar across all 3 types of memories (Figure 1.4F, left group). After associative training, the latency of the direct memory recall was substantially reduced and the intensity of response was increased (Figure 1.4E-F, middle group). The latency and the response amplitude of the indirect memory were also improved, but the latency was not significantly different from that of response for incorrect memories, and the amplitude was not as strong as for direct memory. Importantly, after sleep, the latency of the indirect memory recall was significantly reduced compare to incorrect one (Figure 1.4E, right group, $t(1163)=24.27$, $p=3.039\text{x}10^{\wedge\text{-}100}$, two-sided T-test) and the intensity of response was significantly increased (Figure 1.4F, right group, $t(319) = -9.64$, $p=2.41\text{x}10^{-19}$). This behavioral change in the network response dynamics highlights the increase in strength of the indirect memory following SWS.

1.4.4 Sleep Increases Modularity of Each Triplet in Layer Two Recurrent Connections

To determine which network changes were responsible for improving indirect relational memories after sleep, we analyzed the changes in synaptic weights. There were two types of plastic connections in the model: feedforward connections between layer 1 and layer 2, and recurrent connections within layer 2. In the feedforward connectivity matrices, we observed that sleep leads to a significant increase in the synaptic input coming from both indirect (Figure 1.5A, right, e.g. connection A to C', $t=-6.98$, $p=1.39 \times 10^{-95}$, two-sided t-test) and direct neuronal groups (Figure 1.5A, right, e.g. connection A to B', $t=-5.66$, $p=5.29 \times 10^{-79}$, two-sided t-test). Importantly, the incorrect memory weights (e.g., X to A') were not significantly greater than their pre-training values (in fact they were smaller than their pre-training values, $p < 1 \times 10^{-100}$, one-sided t-test), suggesting that sleep does not just increase all the connections but only connections related to associated memory items. In the recurrent weights (Figure 1.5B), a similar effect was observed where synaptic input from direct and indirect memory groups was significantly increased to specific neurons after sleep ($p=6.18 \times 10^{-62}$, $p=7.67 \times 10^{-88}$ for both groups (direct and indirect, respectively), two-sided t-test). Interestingly (also see discussion below), synaptic input from a neuronal group to its indirect triplet pair (e.g., A' to C') in the second layer became even larger than the synaptic input from an indirect group in the first layer (e.g., A to C', $p=0.02$, two-sided t-test, average feedforward synaptic input=2.08, average recurrent synaptic input=2.75).

To better quantify changes in the recurrent connections in layer 2, we built and analyzed a graph of 10 nodes, where each node represents a group of 10 neurons (i.e., group A' = 11-20, B'=21-30, …, Z'=71-80) (Figure 1.6, A-D). We created an edge between two groups if there were any strong enough weights (i.e., exceeding a threshold) between these groups (the weight threshold was set at 80% of the maximum weight value at different time points, e.g., threshold before training

= 0.0218, threshold after supervised training = 0.1295, threshold after associative training = 0.1857, threshold after sleep = 0.1913). On the graph, the thickness of the edge depicts how many such weights existed between the two nodes. After supervised training, recurrent weights within trained groups (e.g., between all the neurons from group A') increased, but weights between groups remained weak and the graph was essentially disconnected (Figure 1.6B). After associative training, relatively weak connections were formed between the linking group B' (or Y') and the other relevant groups, A' and C' (or X' and Z') (Figure 1.6C). In addition, the self-connections (recurrent connections within a group) were magnified. Finally, after sleep, the overall connectivity between the group triplets was increased, with weak connections between direct memory pairs becoming stronger (e.g., X'-Y') and new connections forming between indirect memories (e.g., X'-Z') (Figure 1.6D). Overall, these changes suggest that items in each triplet (e.g., X'-Y'-Z') becomes strongly connected to the other items in that triplet so that activation of any one group can lead to activation of the other groups. Thus, after sleep all the neurons in the second layer associated with the items belonging to the same relational memory triplet formed an attractor in synaptic weights space.

To further test this idea, we performed modularity analysis on the time-dependent recurrent weight matrix to determine how clusters of neurons change over the course of training and sleep (see Methods and Materials for details). We used a time-dependent community detection algorithm in order to assign each of the 100 neurons in layer 2 to a community (where community assignment can change over time) based on the synaptic connectivity matrix (Leicht and Newman 2008; Jeub et al. 2020). Figure 1.6E illustrates how the community assignment changed during supervised training, associative training, and sleep. During supervised training, each of the 6 subgroups was put into a community with itself, as the neurons within these groups became strongly

interconnected. During associative training, there was some mixing between these six subgroups, as observed, e.g., in the merging of communities representing Y' and Z' (orange group in Figure 1.6E). Finally, during sleep, we observed merging of each of the three subgroups from each triplet into larger community. We found that the number of communities in the network started out high but was further reduced mostly during associative training (Figure 1.6F). Together, these results suggest that sleep altered the connectivity matrix to enable formation of a large community of related neurons who all shared similar stimulus-response profiles - formation of indirect memories. Thus, sleep altered the community structure by building a strong attractor among members of each of the memory triplets.

1.4.5 Replay During Sleep Drives Synaptic Weight Changes

Given that during sleep synaptic weights are restructured to support formation of indirect associative memory, the question remains of what it is specifically about sleep that leads to these changes. Based on our previous work (Wei, Krishnan, and Bazhenov 2016; González et al. 2020; Wei et al. 2018), we hypothesized that replay during sleep of synaptic traces formed during training leads to a strengthening of these synaptic traces and thus an improvement in memory (Lewis and Durrant 2011; Ji and Wilson 2007). Importantly, since in our model indirect connections, e.g., from A to C' or A' to C', are never explicitly activated during training, these pathways may become active during SWS, which could explain the weight changes illustrated above.

To detect possible replay events, we applied a procedure previously proposed in (Gonzalez et al, 2020). After detecting individual Up states (using LFP thresholding, see Methods, Figure 1.7A), we identified, for each Up state, all spiking events that could lead to STDP changes. Thus, if Neuron I fired during an Up state and this was followed by Neuron II firing (within a 200ms time window), then this pair was considered an STDP event and the direction of replay (from

Neuron I to Neuron II) was recorded. We observed that the number of STDP events within the trained region of the network, both in feedforward and recurrent connections, was significantly greater than outside of the trained regions (Figure 1.7B, $p<1e-5$, for visualization purposes, only pairs with number of replay events above a threshold (top 75%) are shown). Importantly, we observed not just more STDP events randomly distributed across all the neuronal pairs in the trained region, but a higher number of events in specific neuronal pairs (Figure 1.7B, note red dots in the regions of interest), suggesting that those events reflect replay of the memory elements formed during associative training. In other words, during an Up state, there was a significantly higher chance that the neurons within the trained region would spike in a defined order compared to the neurons outside of the trained region, indicating that SWS does in fact reactivate synaptic memory traces learned during the associative phase.

We next measured the extent to which replay is correlated with synaptic connectivity changes. Thus, we plotted observed synaptic weight change against the total number of replay events per neuronal pair and discovered a significant correlation between the number of replay events for a given connection and the amplitude of the weight change in this connection (Figure 1.7C). This was true for both feedforward and recurrent connections ($R^2=0.62$, $p=1 \times 10^{-12}$ for feedforward and $R^2=0.41$, $p=1 \times 10^{-10}$ for recurrent connections). These data suggest that sleep replay can restructure weights to build the communities underlying relational memory formation as reported in Figure 1.6. We next separated replay events based on the type of connection: either self-connection (e.g., A-A', or A'-A'), direct connection (e.g., A-B', A'-B'), indirect connection (e.g., A-C', A'-C'), or incorrect connection (e.g., A-X', A'-X'). In feedforward connections, we observed that self-connections had the largest number of replay events, followed by direct, indirect, and incorrect connections, in order (Figure 1.7D, top, number of replay events is averaged

across ten trials and all the connections in each of the four categories). This suggests that in feedforward connections, replay reflects the underlying strength of the synaptic weights (compared Fig. 1.7D top and Fig. 1.5A). Since self-connections were the strongest (Figure 1.5A, t=-3.99, $p=6.72x10^{-5}$, two-sided t-test), these connections experienced the greatest number of replay events. However, in the recurrent connections, there was a greater amount of replay events in the indirect connections (Figure 1.7D, bottom, t=2.72, p =0.006, two-sided t-test). This type of replay can lead to the formation of the communities (Figure 1.6), responsible for formation of indirect associative memories.

1.4.6 N3 Sleep is Uniquely Responsible for Post-sleep Improvement Although Spindle-Slow—Wave Nesting may be Important

Behavioral studies suggest that duration of N3 sleep, but not N2 sleep, during a daytime nap is significantly correlated with associative memory performance (Tucker, Fishbein, and Lau 2010). We tested the effect of N2 sleep by modifying level of neuromodulators in the model, that was set in between their waking and N3 state levels (Krishnan et al. 2016). In this regime, the network generated frequent spindle events interrupted by occasional slow waves (Figure 1.8A). We compared four conditions: 300s of N3 sleep alone (control, as in above simulations), 300s of N2 sleep alone, 600s of N2 sleep alone, and 300s of mixed sleep (200s of N2 followed by 100s of N3). We found that N2 sleep alone was not sufficient to significantly boost associative memory performance, for either 300s or 600s of N2 sleep duration ($t(9) = -1.56$, $p = 0.13$, one-sided t-test) (Figure 1.8B, left). However, either 300s of N3 sleep or mixed N3+N2 sleep did result in a significant improvement ($t(9) = -2.39$, $p = 0.028$, one-sided t-test) (Figure 1.8B, right). These results confirm behavioral evidence showing a unique role for N3 sleep, as opposed to N2 sleep, in improving relational memory.

Other studies suggested that phase locking between slow-waves and spindles (frequency nesting) may be necessary for memory consolidation (Latchoumane et al. 2017; Kim, Gulati, and Ganguly 2019). We tested this by measuring the power in spindle frequency band (from 7 to 16 Hz) in 3 distinct phases of the N3 slow oscillations: Down to Up transition, Up to Down transition, and Random time windows during the Up state. Local field potentials were computed and the starts and ends of each Up state were identified as done previously (see Methods). We calculated the spindle power in 100ms time windows centered in each of the 3 phases. We found significantly higher power in the spindle frequency band near the Down to Up transition compared to the two other phases tested (Figure 1.8D). Additionally, we found that this spindle power was significantly correlated with associative memory improvement following sleep (Figure 1.8C, $R^2 = 0.5$, $p = 0.03$). These results predict that phase-locking between spindles and slow waves may be important in relational memory.

1.5 Discussion

How does sleep give rise to relational memory? Our study suggests the following conceptual model. First, for each "basic" memory, there exists a feedforward pathway through the network that is stable and robust, so a stimulus presentation, i.e., pattern activation in primary sensory area (e.g., neuron group A, Figure 1.9A, left), leads to reliable and unique response in associative cortex (activation of neuron group A'). These pathways can possibly form during development, can be strengthened during subsequent training, and need to be robust for associative learning to take place. These pathways represent sensory "primitives" that have been once learned and do not need to be changed in adult brain. Second, during associative learning, events that have shared context are learned to be represented together. In the model, this occurred when inputs A and B are presented together, which leads to an overlapping representation in associative cortex,

where presentation of A or B alone leads to firing and recollection of the other object (i.e., B' or A') (Figure 1.9A, middle). If different associative memories include a common item, e.g., A-B and B-C, sleep aids in forming indirect associative memory between nonoverlapping items A' and C' by strengthening the entire pathway A → C' (or C →A'), both through an increase in feedforward connections from A to B' and C' as well as community (or attractor) formation for the entire A'-B'-C' group in associative cortex (Figure 1.9A, right). As sleep replay takes place on a compressed timescale (Nádasdy et al. 1999), the entire group (A'-B'-C') can be activated within a small enough window for connections to grow between A' and C', taking advantage of STDP type mechanisms. Indeed, inhibiting the overlapping elements (B/B' or Y/Y') during sleep (or during memory recall) prevents post-sleep improvement on this associative memory task in our model (Figure 1.9B), in line with in vivo work which showed that associations between a visual stimulus and fear response could be blocked by optogenetic inhibition of neurons representing the visual stimulus during sleep (Clawson et al. 2021).

Recent experiments suggest that learning rules may differ between anesthetized and awake states and are biased towards synaptic depression during Up states of SOs in urethane-anesthetized mice (González-Rueda et al. 2018). This result supports the synaptic homeostatic downscaling (SHY) hypothesis suggesting that during sleep synapses are downscaled to free up synaptic resources for learning during the next wake state (Tononi and Cirelli 2014). The other view is that synaptic potentiation occurs during NREM sleep to enable memory consolidation (Igor Timofeev and Chauvette 2018) (see also review in (Puentes-Mestril and Aton 2017). In our new study, based on a large scope of existing experimental data, we utilized a symmetric STDP rule, that is similar in both wake and sleep states, and we observed strengthening of synaptic connections to form new associative memories during sleep. This model may need to be extended based on prevailing

biological views about plasticity rules in the waking and sleeping brain as new data are accumulated. In addition, plasticity mechanisms such as heterosynaptic and homeostatic synaptic plasticity may affect learning and their effects are different between sleep and wake. Indeed, e.g., the effect of heterosynaptic plasticity depends on neuromodulators (Bannon et al. 2017) whose levels fluctuate during sleep-wake cycle. In our new study we explicitly tested effect of heterosynaptic plasticity on associative memory and found that it helps to form associative memories. Because of the complexity of the effects of neuromodulation, we, however, considered simplified model where heterosynaptic scaling operates similarly during sleep and awake.

Our work expands upon computational models of relational memory by providing a biophysically plausible account of learning during waking and consolidation during sleep. Previous models for relational memory include the temporal context model (TCM) and retrieval based models (Kumaran 2012; Kumaran and McClelland 2012). Our model adds to this literature by: 1) developing a biophysical account, based on STDP rules, that explores the role of sleep replay on relational memory tasks, and 2) suggesting a role for both the TCM and retrieval-based models, based on different types of relational memory tasks. TCM and retrieval-based models have been successful at demonstrating performance on associative memory tasks (Kumaran and McClelland 2012). However, these models were constructed using pre-set weights between different regions of the network and sleep replay was implemented using artificial stimulation. In contrast, in our work, we show that STDP rules can be used based on realistic task settings to learn relational memories and synaptic replay, that is needed for formation of indirect relational memories, occurs naturally during SWS and does not require any additional stimulation. We found that, during slow-wave sleep, individual items were replayed spontaneously and in a correct order to form a new relational memory.

Our model, which more closely aligns with TCM, may be insufficient at explaining generalization on ordered relational memory tasks (Ellenbogen et al. 2007; Werchan and Gómez 2013). We showed that replay is as likely to occur in the forward or backward directions (e.g., forward=A → B, backward=B → A). In this simplified task, memory consolidation during sleep occurs mainly in a recurrent layer, as neurons representing single units become wired together based on a shared context and form an attractor or community that enables indirect memory recall. However, in an ordered relational memory task, where the hierarchy of items needs to be learned, replay within a single attractor-based layer may be insufficient to correctly encode the order of the task, and big-loop recurrency may be necessary.

Many studies explored the effect of sleep on relational memory without analyzing correlation between specific sleep stages and performance improvement (Hiuyan Lau, Alger, and Fishbein 2011; Huguet et al. 2019). Our work expands upon these studies by suggesting a unique role for slow-wave sleep in improving relational memory. We further predict that while nesting spindles and slow waves may be important for consolidation of relational memories, spindles alone are not sufficient for consolidation. Our study predicts that the number of slow waves observed during sleep is significantly correlated with the subject's ability to perform relational memory tasks, in line with previous work that demonstrated a significant correlation between the SWS length and relational memory learning (Tucker, Fishbein, and Lau 2010).

Our study also further supports evidence that mental health disorders, such as schizophrenia, where SWS is disrupted may experience deficits in relational memory (Titone et al. 2004; Martin, Jeste, and Ancoli-Israel 2005; Pritchett et al. 2012). Patients with schizophrenia have shown a marked decrease compared to healthy controls in their performance on transitive inference and relational memory tasks (Titone et al. 2004; Avery et al. 2021). One of the deficits

in sleep in schizophrenia subjects is a significant decrease in the amount of SWS (Keshavan, Reynolds, and Kupfer 1990; Yang and Winkelman 2006; Manoach and Stickgold 2009; Benca et al. 1992). Our model suggests that if disrupted SWS is responsible for deficiencies to learn transitive inference in schizophrenia, then methods focusing on recovery of normal sleep patterns in schizophrenia could lead to an improvement in associated cognitive symptoms.

We should note the limitation of our work by ignoring the explicit impact of the hippocampus on memory consolidation and transitive inference. Previous studies have described the importance of the hippocampus in transitive inference tasks, where hippocampal activation is increased during the performance of transitive inference tasks, and damage to the hippocampus decreases performance on such tasks (DeVito, Kanter, and Eichenbaum 2010; Heckers et al. 2004; Wendelken and Bunge 2010; Zalesak and Heckers 2009). Recent studies revealed a complex bi-directional model of the interaction between hippocampal and cortical networks (Helfrich et al. 2019; Rothschild, Eban, and Frank 2017). Our recent modeling work (Sanda et al. 2021) found that hippocampal ripples can coordinate large-scale spatio-temporal dynamics of cortical slow waves. We address these concerns by noting the similarity of the second layer in our model with hippocampal regions, which rely on similar attractor dynamics (Colgin et al. 2010). Thus, the same mechanisms we propose here may explain relational memory improvement during sleep in cortico-hippocampal system. Importantly, empirical and computational studies reported that hippocampal activation during SWS is preceded by cortical input and follows a cortical-hippocampal-cortical pathway (Navarrete, Valderrama, and Lewis 2020; Rothschild, Eban, and Frank 2017; Sanda et al. 2021). In this scenario, the content of replay may be introduced by cortical networks (layer 1 in our model) and lead to the chosen content of replay in hippocampal and other cortical networks (layer 2 in the model).

REM sleep is likely to be very critical in memory and learning but its specific role in formation of relational memories is unknown. One study found that a fraction of time spent in REM sleep during a 60-minute nap was correlated with improvement on A-C item pairs but also led to more forgetting of directly learned (A-B) relations (Alger and Payne 2016). In this work, however, subjects who did not attain REM sleep during the 60-minute period also performed similarly to those that attained REM sleep. Thus, it remains an open question how REM and NREM sleep can differentially contribute to relational memory and to memory consolidation in general (see, however, Wei et al. 2018). It is also likely that the cycling between REM and NREM sleep over the course of a typical night, i.e., multi-phasic sleep with specific temporal structure, is important for sleep-dependent memory consolidation.

To summarize, we built a model of the thalamocortical system which suggests specific biophysical mechanisms that explain the role of sleep in the formation of indirect associative memories. This model predicts that inhibition of neuronal groups that represent items that link associated items may decrease performance on relational memory tasks (Clawson et al. 2021), while artificial stimulation during sleep replay of non-associated items may lead to false memory formation (Diekelmann, Born, and Wagner 2010). Our model can be extended in order to describe transitive inference tasks where there is an underlying hierarchy of items (e.g., A>B) which likely requires a third layer to account for big-loop recurrency needed to perform ordered transitive inference.

1.6 Acknowledgements

This work was supported by ONR (MURI: N00014-16-1-2829), NSF (IIS-1724405) and NIH (1RO1MH125557 and 1RO1MH117155). Chapter 1, in full, is a reprint of the material as it was submitted to The Journal of Neuroscience. Tadros, Timothy; Bazhenov, Maxim. Journal of

Neuroscience, 2022. The dissertation author was one of the primary investigators and first author on this paper.

1.7 Figures

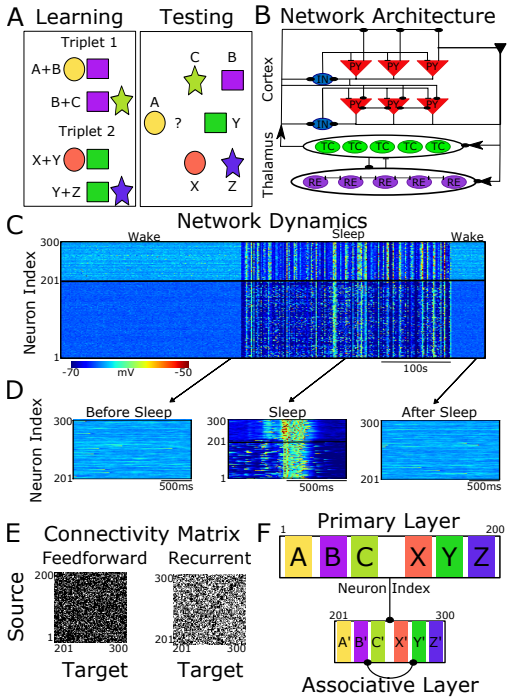

Figure 1.1: Thalamocortical model of relational memory simulates transitions between awake and sleep states.

(A) Basic task setup. During associative training (left), pairs of items are presented simultaneously (A+B, B+C). The relational memory task (right) tests the ability of the network to retrieve direct (B) and indirect (C) items, when presented with item A.

(B) Basic network architecture (PY: Excitatory pyramidal cells; IN: Inhibitory interneurons; TC: Thalamocortical neurons; RE: inhibitory thalamic reticular neurons). Excitatory connections terminate in a dot, whereas inhibitory connections terminate in a line. Arrows indicate the direction of connections.

(C) Baseline network dynamics of the 200 PY neurons and 100 INs during wake and slow-wave sleep (each row depicts membrane potential over time of a single neuron).

(D) Zoom-in of baseline network dynamics in awake state before sleep (left), during sleep (middle; one Up state is shown), and in awake state after sleep (right). Network dynamics before and after sleep are shown for layer two neurons. During sleep, a canonical slow wave pattern is seen across both layers.

(E) Weight connectivity matrix for feedforward connections from layer 1 to layer 2 in cortex (left) and recurrent connections within layer 2 (right). Connection probability is 30% for feedforward connections and 50% for recurrent connections. A white dot represents that a connection exists between two neurons.

(F) Two-layer cortical network architecture. There are plastic feedforward connections from layer 1 to layer 2 and plastic recurrent connections within layer 2. A subset of neurons in each layer is trained to represent individual items (e.g., neurons 10-29 (denoted neuron group A in the text) in layer 1 represent item A and neurons 210-219 (denoted neuron group A') represent item A in the second layer).

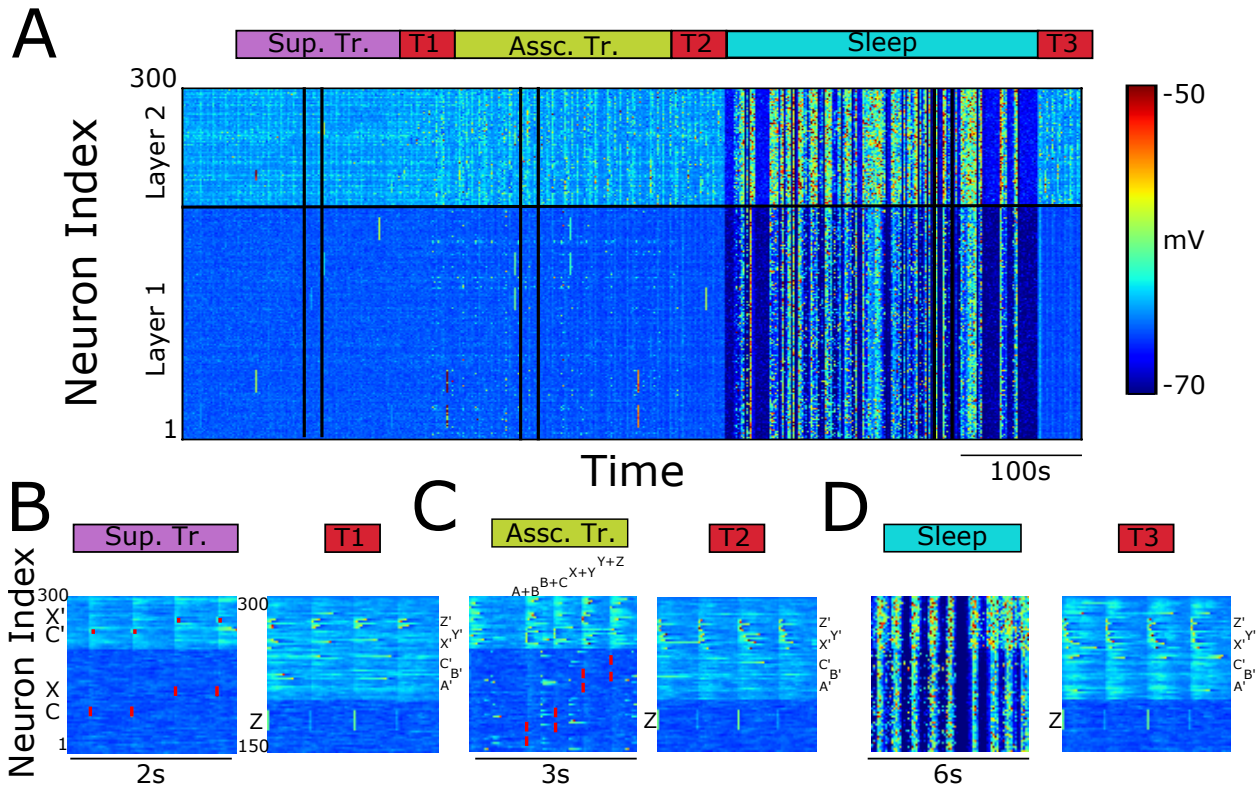

Figure 1.2: Training and testing protocol include supervised and associative training in awake state and spontaneous activity during SWS.

(A) Overall network dynamics for the three phases: supervised training (purple), associative training (green) and sleep (cyan). Each phase is followed by a testing phase (T1, T2, and T3).

(B) During supervised training, neuron groups A, B, C, X, Y, Z are stimulated in layer 1 and neuron groups A', B', C', X', Y', Z', respectively, are stimulated in layer 2 with a 5 ms time delay. Example stimulations of C and C' and X and X' are shown on the left. During testing, a single neuron group in layer 1 is stimulated (e.g. neuron group Z on the right), and the response of neurons in layer 2 are measured. Red bars are shown to accentuate neuron groups that are stimulated.

(C) During associative training, neuron groups A+B, B+C, X+Y, Y+Z are stimulated simultaneously. Each pair is stimulated with a 500 ms delay after previous group stimulation. No stimulation is provided in layer 2. After associative training, another testing phase is performed.

(D) During sleep, neuromodulator levels are altered in order to simulate deep stage 3 (N3) sleep activity characterized by spontaneous slow-waves across cortex. After sleep, another testing phase is performed.

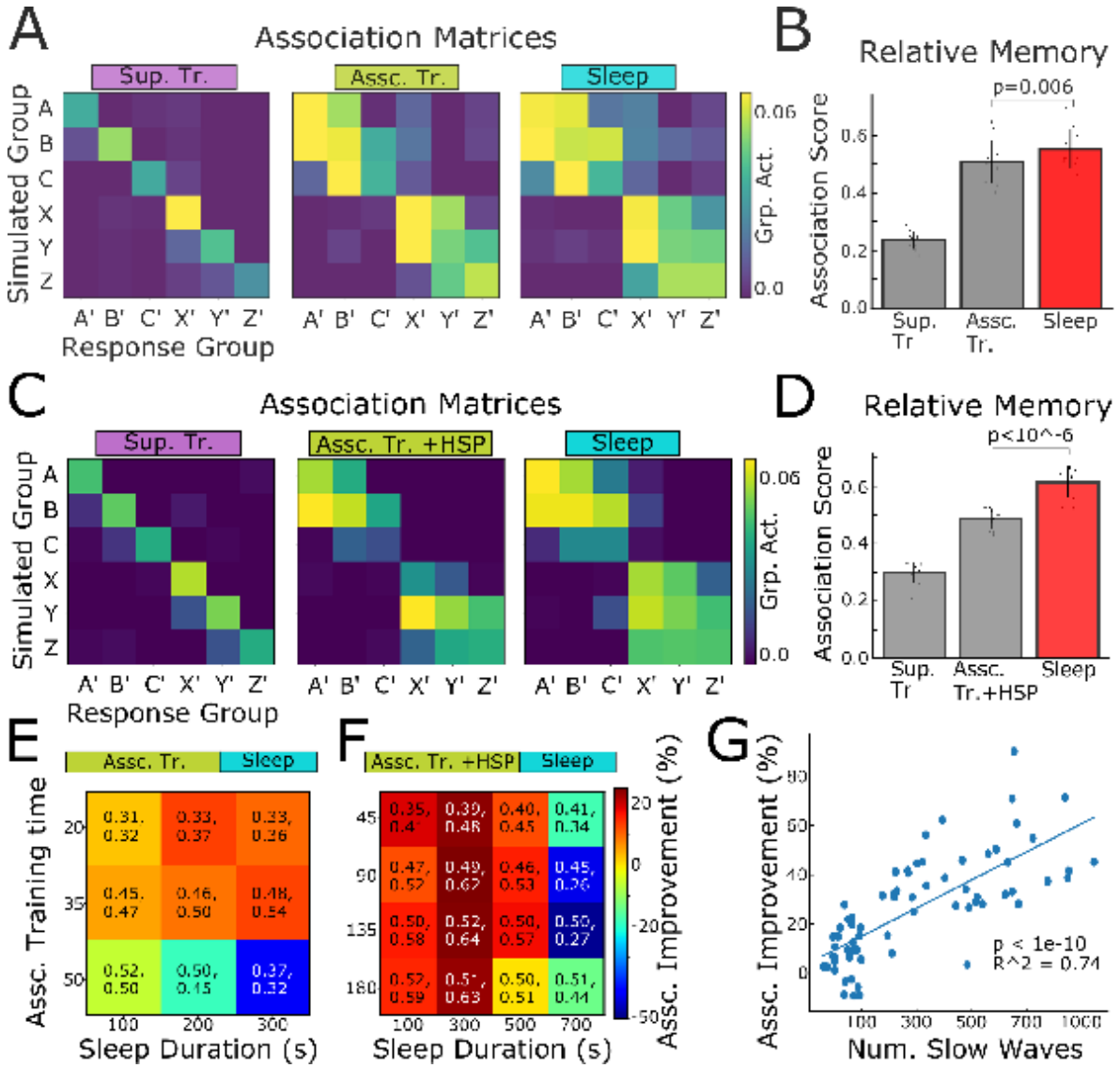

Figure 1.3: Sleep improves associative memory performance

(A,C) Responses of layer 2 neuron groups after stimulating a neuron group in layer 1 during testing after supervised training (left), associative training (middle), and sleep (right). Panel (A) depicts responses in the model without heterosynaptic plasticity (HSP) and panel (C) is from the model with heterosynaptic plasticity included during associative training.
(B,D) Conversion of association matrices shown in (A) and (C) to a single association performance score. Panel (B) is without heterosynaptic plasticity and panel (D) is with heterosynaptic plasticity.
(E,F) Associative training duration vs. sleep duration. Panel (E) is from the model without heterosynaptic plasticity and panel (F) is from the model with heterosynaptic plasticity. The first number in each cell depicts the association score before sleep and the second number depicts the association score after sleep. Color depicts the % change in association score from after to before sleep.
(G) Improvement in association score as a function of number of slow waves ($p=2.45 \times 10^{-13}$, $R^2=0.74$) in the model including heterosynaptic plasticity. Each dot represents a different network trial. Network trials are computed for 100s, 300s, and 500s of sleep as well as different durations of associative training.

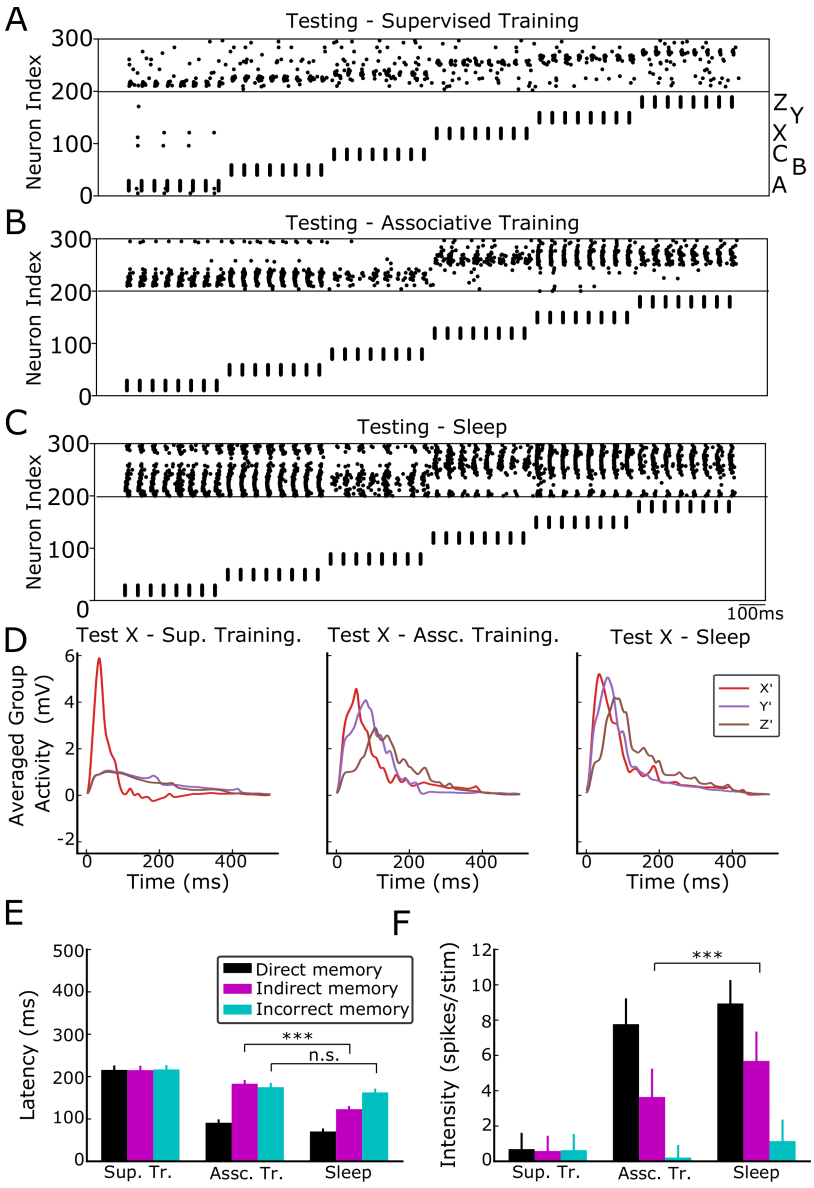

Figure 1.4: Sleep increases amplitude and decreases latency of indirect memory response.

(A,B,C) Raw network response traces during testing phase of stimulating A, B, C, X, Y, Z (from left to right) after supervised training (A), associative training (B), and sleep (C). Note increase in response and decrease in latency after sleep.
(D) Averaged (across 8 trials) and smoothed, through a band-pass filter at 0.1 and 20 Hz, local field potential (LFP) computed separately for the three neuron groups in layer 2 (X', Y', Z' are shown) upon simulations of a neuron group X in layer 1. LFPs are shown during testing phase after supervised training (left), associative training (middle), and sleep (right).
(E) Average response latency for direct memories (black, e.g., latency of neuron group B' when A is stimulated), indirect memories (pink, e.g. latency of neuron group C' when A is stimulated), and incorrect memories (cyan, e.g. latency of neuron group X' when A is stimulated).
(F) Average firing rate of neurons in layer 2 for each type of memory (direct, indirect, and incorrect) during testing phase.

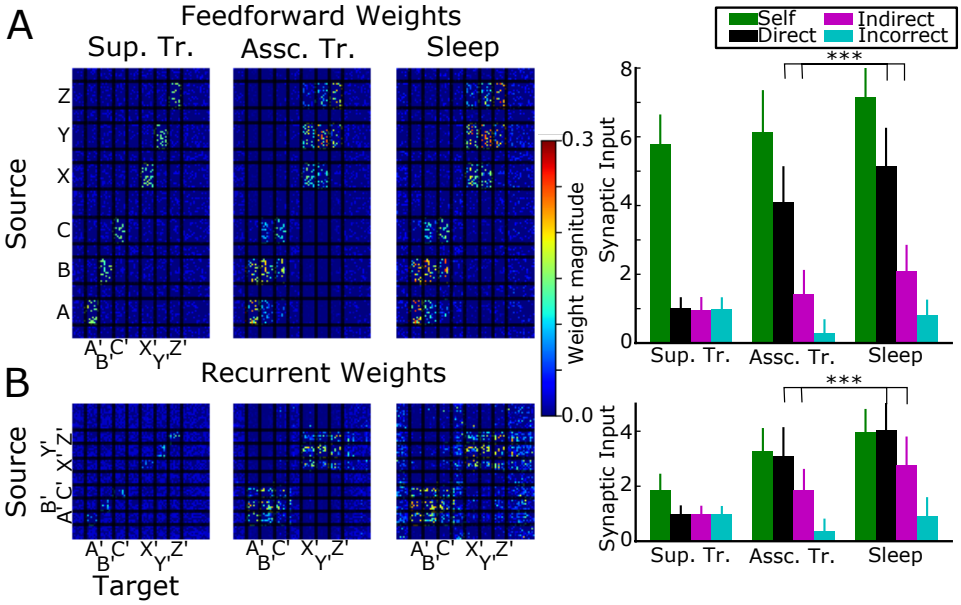

Figure 1.5: Synaptic weight dynamics explains improvements in relational memory after sleep.

(A,B) Left, Feedforward (A) and recurrent (B) synaptic weight matrices after supervised training, associative training, and sleep. Right, Synaptic input to the neurons of each memory type in layer 2 (the sum of all the weights connecting to those neurons) for self-memories (A-A'), direct memories (A-B'), indirect memories (A-C'), and incorrect memories (A-X') after supervised training, associative training, and sleep.

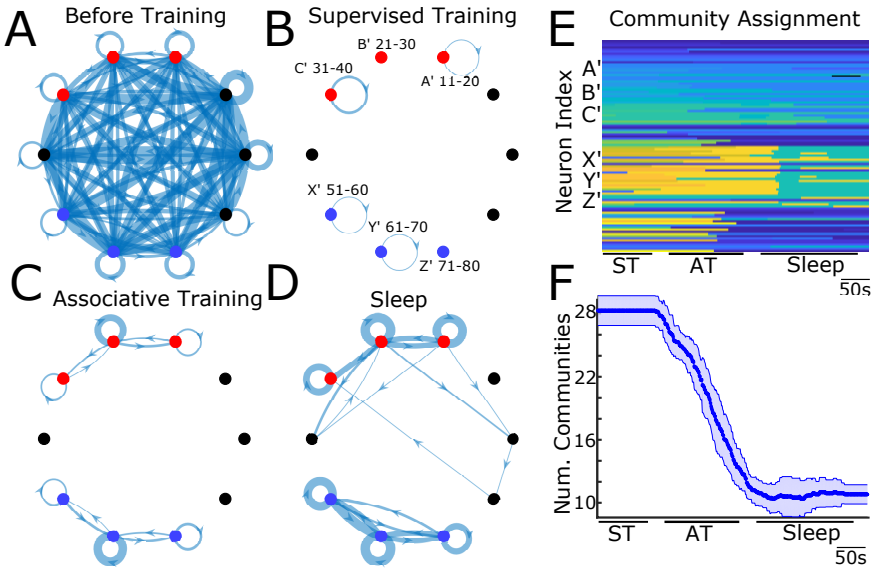

Figure 1.6: Sleep increases modularity of each item triplet (A'B'C' and X'Y'Z') in layer 2 recurrent connections.

(A-D) Graphs of layer 2 connectivity matrices. Each dot represents a group of 10 neurons (red dots = A', B', C', blue dots = X',Y',Z'). A line is drawn between two dots if there is a weight between groups that exceeds a given threshold (75% of the maximal wight). The thickness of the line represents the number of such connections. **(A)** before any training, **(B)** after supervised training, **(C)** after associative training, and **(D)** after sleep. Note that threshold is calculated for each state separately, so, e.g., before training many connections exceed the threshold defined by initial weak connections.
(E) Community assignment for layer 2 neurons over time during each training/sleep phase (ST = supervised training, AT = associative training, and sleep). Neurons were assigned the same color (at any given time) if those neurons belonged to the same community.
(F) The number of communities over time. Data are averaged across 10 network trials and error bars indicate the standard deviation across trials.

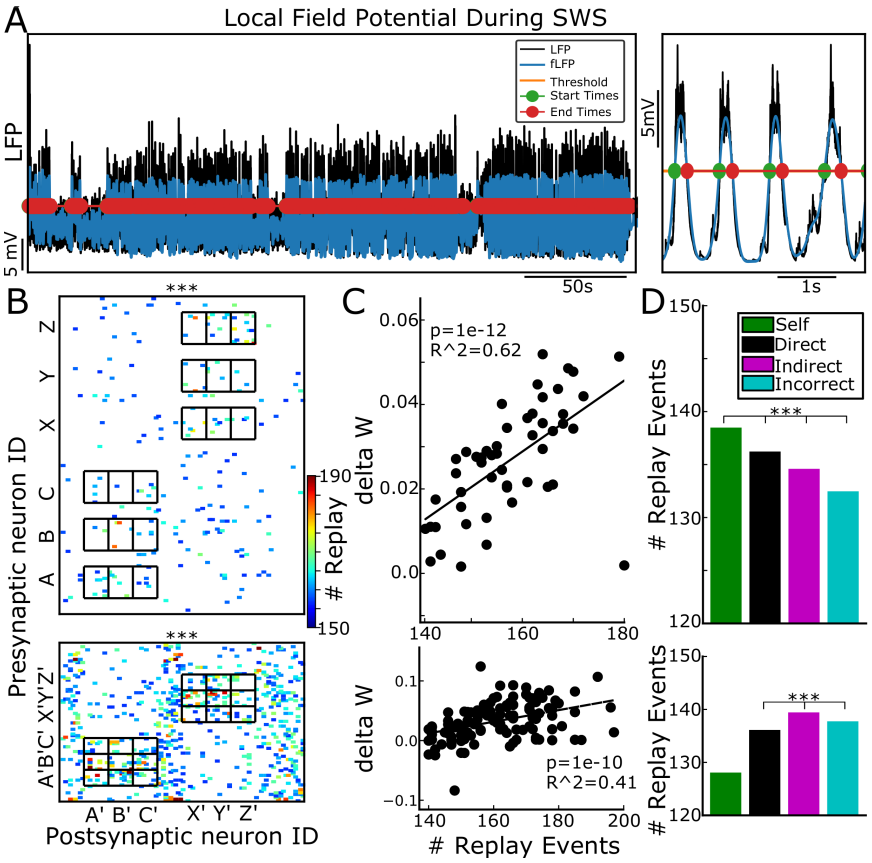

Figure 1.7: Replay during sleep drives synaptic weight changes.

(A) Local field potential during SWS (left) and examples of slow-waves (right). Beginning/end times of Up and Down states are computed by setting a threshold for the transition from Down to Up state and vice versa.
(B) Number of replay events for feedforward (top) and recurrent (bottom) connections. Replay events are selected by identifying sequential ordered firing events, within a specified time window. Replay events occur significantly more in the areas of interest (black grids) than in other areas ($p<1e-4$, based on shuffling replay matrix 10,000 times).
(C) Change in synaptic weights as a function of number of replay events between neurons for feedforward (top, $R2 = 0.61$, $p=1 \times 10^{-12}$) and recurrent (bottom, $R2=0.41$, $p=1x\ 10^{-10}$) connections.
(D) Number of replay events between self, direct, indirect, and incorrect neuron groups for feedforward (top) and recurrent (bottom) connections. For feedforward connections, there was a significantly higher number of replay events between self-connections than direct connections, direct connections than indirect connections, and indirect connections than incorrect connections. For recurrent connections, indirect connections revealed the most replay events ($p=0.006$ between wrong connections, and $p=3.28 \times 10^{-36}$ between direct connections).

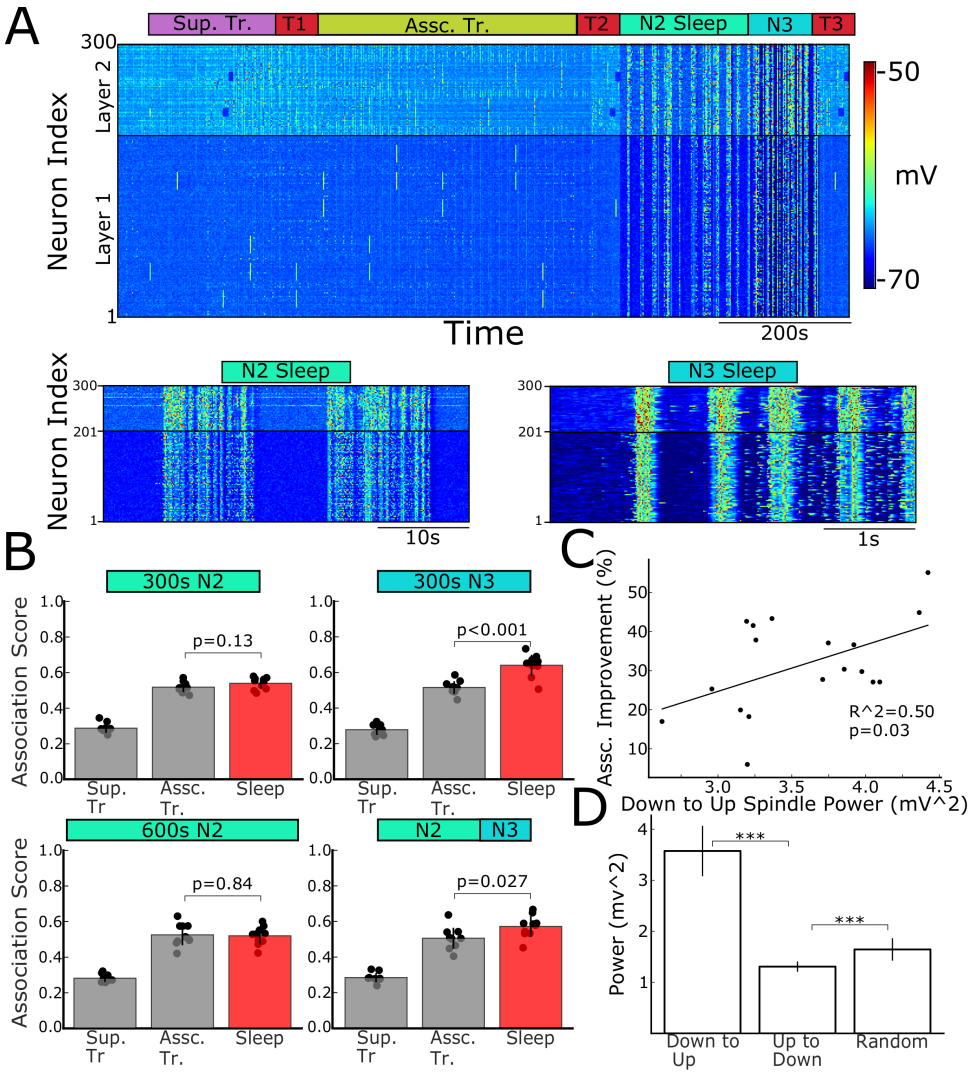

Figure 1.8: State 2 (N2) sleep has little effect on association score, although spindle/slow oscillation nesting during N3 sleep reveals significance.

(A) Network dynamics including both N2 and N3 sleep: supervised training (purple), associative training (green) and sleep, comprised of N2 (lime) and N3 sleep (cyan). Bottom row shows zoom-in of N2 sleep (two spindles are shown) and N3 sleep (slow-waves)

(B) Association scores following 300s of N2 sleep (top left), 300s of N3 sleep (top right), 600s of N2 sleep (bottom left), and 300s of mixed sleep (200s N2 and 100s N3, bottom right).

(C) Association score improvement as a function of spindle power near Down-to-Up transition of N3 sleep suggests a significant correlation between spindle/slow oscillation nesting and association score. Spindle power in 1000s of mV^2.

(D) Spindle power is significantly higher near Down-to-Up transition then near Up-to-Down transition or a random time selected during the Up state of a slow-wave. Power was calculated based on 100ms time windows.

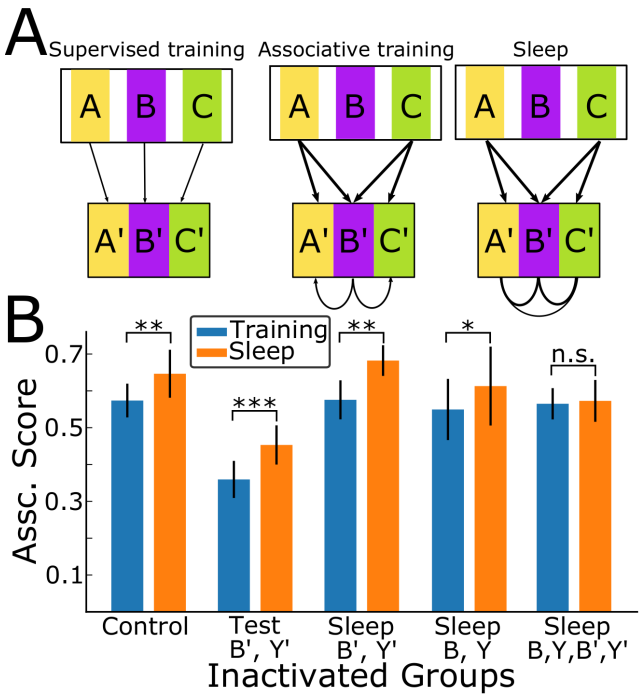

Figure 1.9: Proposed model of relational memory and main experimental predictions.

(A) Summary of the changes to the model at different time points. During supervised training, feedforward connections are formed between layers 1 and 2 to represent self-memories (e.g., A-A'). During associative training, the network learns to associate items presented together (e.g., A with B and B with C). However, these connections are weak and no indirect associations are learned (e.g., A is not associated with C). After sleep, direct and indirect memory connections are strengthened and one attractor is formed for entire triplet of items, i.e., a community including A', B', and C'.

(B) Effect of inactivating different neuronal groups during either sleep or testing on association score. Blue bars show performance after training and orange ones show performance after sleep. Silencing linking group in any one layer only (B' or B, Y' or Y) during sleep still leads to significant post-sleep improvement for associative memories (B', Y' - $t(10) = -4.91$, $p = 0.001$; B, Y - $t(10) = -2.03$, $p = 0.045$, one-sided t-test, FDR correction). However, silencing linking groups in both layers (B/B', Y/Y') during sleep prevents post-sleep improvement for these associative memory tasks ($t(10) = -0.59$, $p = 0.28$). Inactivating linking groups in layer 2 alone (B', Y') during testing was sufficient to significantly reduce associative memory performance.

1.8 References

Alexander-Bloch, Aaron F., Nitin Gogtay, David Meunier, Rasmus Birn, Liv Clasen, Francois Lalonde, Rhoshel Lenroot, Jay Giedd, and Edward T. Bullmore. 2010. "Disrupted Modularity and Local Connectivity of Brain Functional Networks in Childhood-Onset Schizophrenia." *Frontiers in Systems Neuroscience* 4. https://doi.org/10.3389/fnsys.2010.00147.

Alger, Sara E., and Jessica D. Payne. 2016. "The Differential Effects of Emotional Salience on Direct Associative and Relational Memory during a Nap." *Cognitive, Affective, & Behavioral Neuroscience* 16 (6): 1150–63. https://doi.org/10.3758/s13415-016-0460-1.

Aminoff, Elissa M., and Michael J. Tarr. 2015. "Associative Processing Is Inherent in Scene Perception." *PLOS ONE* 10 (6): e0128840. https://doi.org/10.1371/journal.pone.0128840.

Avery, Suzanne N, Kristan Armstrong, Maureen McHugo, Simon Vandekar, Jennifer Urbano Blackford, Neil D Woodward, and Stephan Heckers. 2021. "Relational Memory in the Early Stage of Psychosis: A 2-Year Follow-up Study." *Schizophrenia Bulletin* 47 (1): 75–86. https://doi.org/10.1093/schbul/sbaa081.

Bannon, Nicholas M., Marina Chistiakova, Jen-Yung Chen, Maxim Bazhenov, and Maxim Volgushev. 2017. "Adenosine Shifts Plasticity Regimes between Associative and Homeostatic by Modulating Heterosynaptic Changes." *Journal of Neuroscience* 37 (6): 1439–52. https://doi.org/10.1523/JNEUROSCI.2984-16.2016.

Bassett, Danielle S., Muzhi Yang, Nicholas F. Wymbs, and Scott T. Grafton. 2015. "Learning-Induced Autonomy of Sensorimotor Systems." *Nature Neuroscience* 18 (5): 744–51. https://doi.org/10.1038/nn.3993.

Benca, Ruth M., William H. Obermeyer, Ronald A. Thisted, and J. Christian Gillin. 1992. "Sleep and Psychiatric Disorders: A Meta-Analysis." *Archives of General Psychiatry* 49 (8): 651–68. https://doi.org/10.1001/archpsyc.1992.01820080059010.

Bjekić, Jovana, Marija V. Čolić, Marko Živanović, Sladjan D. Milanović, and Saša R. Filipović. 2019. "Transcranial Direct Current Stimulation (TDCS) over Parietal Cortex Improves Associative Memory." *Neurobiology of Learning and Memory* 157 (January): 114–20. https://doi.org/10.1016/j.nlm.2018.12.007.

Blake, H., and R. W. Gerard. 1937. "BRAIN POTENTIALS DURING SLEEP." *American Journal of Physiology-Legacy Content* 119 (4): 692–703. https://doi.org/10.1152/ajplegacy.1937.119.4.692.

Chatburn, Alex, Kurt Lushington, and Mark J. Kohler. 2014. "Complex Associative Memory Processing and Sleep: A Systematic Review and Meta-Analysis of Behavioural Evidence and Underlying EEG Mechanisms." *Neuroscience & Biobehavioral Reviews* 47 (November): 646–55. https://doi.org/10.1016/j.neubiorev.2014.10.018.

Chen, Jen-Yung, Peter Lonjers, Christopher Lee, Marina Chistiakova, Maxim Volgushev, and Maxim Bazhenov. 2013. "Heterosynaptic Plasticity Prevents Runaway Synaptic Dynamics." *Journal of Neuroscience* 33 (40): 15915–29. https://doi.org/10.1523/JNEUROSCI.5088-12.2013.

Chistiakova, Marina, Nicholas Bannon, Jen-Yung Chen, Maxim Bazhenov, and Maxim Volgushev. 2015. "Homeostatic Role of Heterosynaptic Plasticity: Models and Experiments." *Frontiers in Computational Neuroscience* 9: 89. https://doi.org/10.3389/fncom.2015.00089.

Chistiakova, Marina, Nicholas M. Bannon, Maxim Bazhenov, and Maxim Volgushev. 2014. "Heterosynaptic Plasticity: Multiple Mechanisms and Multiple Roles." *The Neuroscientist* 20 (5): 483–98. https://doi.org/10.1177/1073858414529829.

Clawson, Brittany C., Emily J. Pickup, Amy Ensing, Laura Geneseo, James Shaver, John Gonzalez-Amoretti, Meiling Zhao, et al. 2021. "Causal Role for Sleep-Dependent Reactivation of Learning-Activated Sensory Ensembles for Fear Memory Consolidation." *Nature Communications* 12 (1): 1200. https://doi.org/10.1038/s41467-021-21471-2.

Colgin, Laura L., Stefan Leutgeb, Karel Jezek, Jill K. Leutgeb, Edvard I. Moser, Bruce L. McNaughton, and May-Britt Moser. 2010. "Attractor-Map Versus Autoassociation Based Attractor Dynamics in the Hippocampal Network." *Journal of Neurophysiology* 104 (1): 35–50. https://doi.org/10.1152/jn.00202.2010.

De Meo, Pasquale, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. 2011. "Generalized Louvain Method for Community Detection in Large Networks." *2011 11th International Conference on Intelligent Systems Design and Applications*, November, 88–93. https://doi.org/10.1109/ISDA.2011.6121636.

DeVito, Loren M., Benjamin R. Kanter, and Howard Eichenbaum. 2010. "The Hippocampus Contributes to Memory Expression during Transitive Inference in Mice." *Hippocampus* 20 (1): 208–17.

Diekelmann, Susanne, and Jan Born. 2010. "The Memory Function of Sleep." *Nature Reviews Neuroscience* 11 (2): 114–26. https://doi.org/10.1038/nrn2762.

Diekelmann, Susanne, Jan Born, and Ullrich Wagner. 2010. "Sleep Enhances False Memories Depending on General Memory Performance." *Behavioural Brain Research* 208 (2): 425–29.

Ellenbogen, Jeffrey M., Peter T. Hu, Jessica D. Payne, Debra Titone, and Matthew P. Walker. 2007. "Human Relational Memory Requires Time and Sleep." *Proceedings of the National Academy of Sciences* 104 (18): 7723–28.

Fitzgerald, Jamie K., David J. Freedman, and John A. Assad. 2011. "Generalized Associative Representations in Parietal Cortex." *Nature Neuroscience* 14 (8): 1075–79. https://doi.org/10.1038/nn.2878.

Fitzgerald, Jamie K., David J. Freedman, Alessandra Fanini, Sharath Bennur, Joshua I. Gold, and John A. Assad. 2013. "Biased Associative Representations in Parietal Cortex." *Neuron* 77 (1): 180–91. https://doi.org/10.1016/j.neuron.2012.11.014.

González, Oscar C, Yury Sokolov, Giri P Krishnan, Jean Erik Delanois, and Maxim Bazhenov. 2020. "Can Sleep Protect Memories from Catastrophic Forgetting?" Edited by Mark CW van Rossum, Laura L Colgin, and Francesco P Battaglia. *ELife* 9 (August): e51005. https://doi.org/10.7554/eLife.51005.

González-Rueda, Ana, Victor Pedrosa, Rachael C. Feord, Claudia Clopath, and Ole Paulsen. 2018. "Activity-Dependent Downscaling of Subthreshold Synaptic Inputs during Slow-Wave-Sleep-like Activity In Vivo." *Neuron* 97 (6): 1244-1252.e5. https://doi.org/10.1016/j.neuron.2018.01.047.

Heckers, Stephan, Martin Zalesak, Anthony P. Weiss, Tali Ditman, and Debra Titone. 2004. "Hippocampal Activation during Transitive Inference in Humans." *Hippocampus* 14 (2): 153–62.

Helfrich, Randolph F., Janna D. Lendner, Bryce A. Mander, Heriberto Guillen, Michelle Paff, Lilit Mnatsakanyan, Sumeet Vadera, Matthew P. Walker, Jack J. Lin, and Robert T. Knight. 2019. "Bidirectional Prefrontal-Hippocampal Dynamics Organize Information Transfer during Sleep in Humans." *Nature Communications* 10 (August): 3572. https://doi.org/10.1038/s41467-019-11444-x.

Hiuyan Lau, Sara E. Alger, and William Fishbein. 2011. "Relational Memory: A Daytime Nap Facilitates the Abstraction of General Concepts." *PloS One* 6 (11): e27139.

Hodgkin, A. L., and A. F. Huxley. 1952. "A Quantitative Description of Membrane Current and Its Application to Conduction and Excitation in Nerve." *The Journal of Physiology* 117 (4): 500–544.

Huguet, Makenzie, Jessica D. Payne, Sara Y. Kim, and Sara E. Alger. 2019. "Overnight Sleep Benefits Both Neutral and Negative Direct Associative and Relational Memory." *Cognitive, Affective, & Behavioral Neuroscience* 19 (6): 1391–1403. https://doi.org/10.3758/s13415-019-00746-8.

Jeub, LG, M Bazzi, IS Jutla, and PJ Muncha. 2020. "A Generalized Louvain Method for Community Detection Implemented in MATLAB. 2011-2017."

Ji, Daoyun, and Matthew A. Wilson. 2007. "Coordinated Memory Replay in the Visual Cortex and Hippocampus during Sleep." *Nature Neuroscience* 10 (1): 100–107. https://doi.org/10.1038/nn1825.

Keshavan, M. S., Charles F. Reynolds, and David J. Kupfer. 1990. "Electroencephalographic Sleep in Schizophrenia: A Critical Review." *Comprehensive Psychiatry* 31 (1): 34–47. https://doi.org/10.1016/0010-440X(90)90052-T.

Kim, Jaekyung, Tanuj Gulati, and Karunesh Ganguly. 2019. “Competing Roles of Slow Oscillations and Delta Waves in Memory Consolidation versus Forgetting.” *Cell* 179 (2): 514-526.e13. https://doi.org/10.1016/j.cell.2019.08.040.

Klinzing, Jens G., Niels Niethard, and Jan Born. 2019. “Mechanisms of Systems Memory Consolidation during Sleep.” *Nature Neuroscience* 22 (10): 1598–1610.

Krishnan, Giri P, Sylvain Chauvette, Isaac Shamie, Sara Soltani, Igor Timofeev, Sydney S Cash, Eric Halgren, and Maxim Bazhenov. 2016. “Cellular and Neurochemical Basis of Sleep Stages in the Thalamocortical Network.” *ELife* 5 (November): e18607. https://doi.org/10.7554/eLife.18607.

Kumaran, Dharshan. 2012. “What Representations and Computations Underpin the Contribution of the Hippocampus to Generalization and Inference?” *Frontiers in Human Neuroscience* 6: 157.

Kumaran, Dharshan, and James L. McClelland. 2012. “Generalization through the Recurrent Interaction of Episodic Memories: A Model of the Hippocampal System.” *Psychological Review* 119 (3): 573.

Latchoumane, Charles-Francois V., Hong-Viet V. Ngo, Jan Born, and Hee-Sup Shin. 2017. “Thalamic Spindles Promote Memory Formation during Sleep through Triple Phase-Locking of Cortical, Thalamic, and Hippocampal Rhythms.” *Neuron* 95 (2): 424-435.e6. https://doi.org/10.1016/j.neuron.2017.06.025.

Lau, Hiuyan, Sara E. Alger, and William Fishbein. 2011. “Relational Memory: A Daytime Nap Facilitates the Abstraction of General Concepts.” *PLOS ONE* 6 (11): e27139. https://doi.org/10.1371/journal.pone.0027139.

Lau, Hiuyan, M. A. Tucker, and W. Fishbein. 2010. “Daytime Napping: Effects on Human Direct Associative and Relational Memory.” *Neurobiology of Learning and Memory* 93 (4): 554–60. https://doi.org/10.1016/j.nlm.2010.02.003.

Leicht, E. A., and M. E. J. Newman. 2008. “Community Structure in Directed Networks.” *Physical Review Letters* 100 (11): 118703. https://doi.org/10.1103/PhysRevLett.100.118703.

Lemieux, Maxime, Jen-Yung Chen, Peter Lonjers, Maxim Bazhenov, and Igor Timofeev. 2014. “The Impact of Cortical Deafferentation on the Neocortical Slow Oscillation.” *Journal of Neuroscience* 34 (16): 5689–5703. https://doi.org/10.1523/JNEUROSCI.1156-13.2014.

Lewis, Penelope A., and Simon J. Durrant. 2011. “Overlapping Memory Replay during Sleep Builds Cognitive Schemata.” *Trends in Cognitive Sciences* 15 (8): 343–51.

Lewis, Penelope A., Günther Knoblich, and Gina Poe. 2018. “How Memory Replay in Sleep Boosts Creative Problem-Solving.” *Trends in Cognitive Sciences* 22 (6): 491–503.

Malsburg, Chr. von der. 1973. “Self-Organization of Orientation Sensitive Cells in the Striate Cortex.” *Kybernetik* 14 (2): 85–100. https://doi.org/10.1007/BF00288907.

Manoach, Dara S., and Robert Stickgold. 2009. “Does Abnormal Sleep Impair Memory Consolidation in Schizophrenia?” *Frontiers in Human Neuroscience* 3. https://doi.org/10.3389/neuro.09.021.2009.

Maquet, Pierre. 2001. “The Role of Sleep in Learning and Memory.” *Science* 294 (5544): 1048–52.

Martin, Jennifer L., Dilip V. Jeste, and Sonia Ancoli-Israel. 2005. “Older Schizophrenia Patients Have More Disrupted Sleep and Circadian Rhythms than Age-Matched Comparison Subjects.” *Journal of Psychiatric Research* 39 (3): 251–59. https://doi.org/10.1016/j.jpsychires.2004.08.011.

Miller, Kenneth D. 1996. “Synaptic Economics: Competition and Cooperation in Synaptic Plasticity.” *Neuron* 17 (3): 371–74. https://doi.org/10.1016/S0896-6273(00)80169-5.

Miyamoto, D., D. Hirai, C. C. A. Fung, A. Inutsuka, M. Odagawa, T. Suzuki, R. Boehringer, et al. 2016. “Top-down Cortical Input during NREM Sleep Consolidates Perceptual Memory.” *Science* 352 (6291): 1315–18. https://doi.org/10.1126/science.aaf0902.

Mucha, Peter J., Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. 2010. “Community Structure in Time-Dependent, Multiscale, and Multiplex Networks.” *Science* 328 (5980): 876–78. https://doi.org/10.1126/science.1184819.

Nádasdy, Zoltán, Hajime Hirase, András Czurkó, Jozsef Csicsvari, and György Buzsáki. 1999. “Replay and Time Compression of Recurring Spike Sequences in the Hippocampus.” *The Journal of Neuroscience* 19 (21): 9497. https://doi.org/10.1523/JNEUROSCI.19-21-09497.1999.

Navarrete, Miguel, Mario Valderrama, and Penelope A Lewis. 2020. “The Role of Slow-Wave Sleep Rhythms in the Cortical-Hippocampal Loop for Memory Consolidation.” *Current Opinion in Behavioral Sciences*, Understanding memory: Which level of analysis?, 32 (April): 102–10. https://doi.org/10.1016/j.cobeha.2020.02.006.

Nieuwenhuis, Ingrid L C, Vasiliki Folia, Christian Forkstam, Ole Jensen, and Magnus Petersson. 2013. “Sleep Promotes the Extraction of Grammatical Rules.” *PLOS ONE* 8 (6): 10.

Pritchett, David, Katharina Wulff, Peter L. Oliver, David M. Bannerman, Kay E. Davies, Paul J. Harrison, Stuart N. Peirson, and Russell G. Foster. 2012. “Evaluating the Links between Schizophrenia and Sleep and Circadian Rhythm Disruption.” *Journal of Neural Transmission* 119 (10): 1061–75. https://doi.org/10.1007/s00702-012-0817-8.

Puentes-Mestril, Carlos, and Sara J. Aton. 2017. “Linking Network Activity to Synaptic Plasticity during Sleep: Hypotheses and Recent Data.” *Frontiers in Neural Circuits* 11: 61. https://doi.org/10.3389/fncir.2017.00061.

Rothschild, Gideon, Elad Eban, and Loren M. Frank. 2017. “A Cortical–Hippocampal–Cortical Loop of Information Processing during Memory Consolidation.” *Nature Neuroscience* 20 (2): 251–59. https://doi.org/10.1038/nn.4457.

Sanda, Pavel, Paola Malerba, Xi Jiang, Giri P Krishnan, Jorge Gonzalez-Martinez, Eric Halgren, and Maxim Bazhenov. 2021. “Bidirectional Interaction of Hippocampal Ripples and Cortical Slow Waves Leads to Coordinated Spiking Activity During NREM Sleep.” *Cerebral Cortex* 31 (1): 324–40. https://doi.org/10.1093/cercor/bhaa228.

Steriade, M. 2006. “Grouping of Brain Rhythms in Corticothalamic Systems.” *Neuroscience* 137 (4): 1087–1106. https://doi.org/10.1016/j.neuroscience.2005.10.029.

Steriade, M., D. A. McCormick, and T. J. Sejnowski. 1993. “Thalamocortical Oscillations in the Sleeping and Aroused Brain.” *Science* 262 (5134): 679–85. https://doi.org/10.1126/science.8235588.

Studte, Sara, Emma Bridger, and Axel Mecklinger. 2015. “Nap Sleep Preserves Associative but Not Item Memory Performance.” *Neurobiology of Learning and Memory* 120 (April): 84–93. https://doi.org/10.1016/j.nlm.2015.02.012.

Sugisaki, Eriko, Yasuhiro Fukushima, Satoshi Fujii, Yoshihiko Yamazaki, and Takeshi Aihara. 2016. “The Effect of Coactivation of Muscarinic and Nicotinic Acetylcholine Receptors on LTD in the Hippocampal CA1 Network.” *Brain Research* 1649 (October): 44–52. https://doi.org/10.1016/j.brainres.2016.08.024.

Teyler, Timothy J., and Pascal DiScenna. 1986. “The Hippocampal Memory Indexing Theory.” *Behavioral Neuroscience* 100 (2): 147–54. https://doi.org/10.1037/0735-7044.100.2.147.

Timofeev, I., F. Grenier, M. Bazhenov, T. J. Sejnowski, and M. Steriade. 2000. “Origin of Slow Cortical Oscillations in Deafferented Cortical Slabs.” *Cerebral Cortex* 10 (12): 1185–99. https://doi.org/10.1093/cercor/10.12.1185.

Timofeev, Igor, and Sylvain Chauvette. 2018. “Sleep, Anesthesia, and Plasticity.” *Neuron* 97 (6): 1200–1202. https://doi.org/10.1016/j.neuron.2018.03.013.

Titone, Debra, Tali Ditman, Philip S. Holzman, Howard Eichenbaum, and Deborah L. Levy. 2004. “Transitive Inference in Schizophrenia: Impairments in Relational Memory Organization.” *Schizophrenia Research* 68 (2–3): 235–47.

Tononi, Giulio, and Chiara Cirelli. 2014. “Sleep and the Price of Plasticity: From Synaptic and Cellular Homeostasis to Memory Consolidation and Integration.” *Neuron* 81 (1): 12–34. https://doi.org/10.1016/j.neuron.2013.12.025.

Tucker, M. A., W. Fishbein, and Hiuyan Lau. 2010. “Daytime Napping: Effects on Human Direct Associative and Relational Memory.” *Neurobiology of Learning and Memory* 93 (4): 554–60. https://doi.org/10.1016/j.nlm.2010.02.003.

Vanini, Giancarlo, Ralph Lydic, and Helen A. Baghdoyan. 2012. “GABA-to-ACh Ratio in Basal Forebrain and Cerebral Cortex Varies Significantly During Sleep.” *Sleep* 35 (10): 1325–34. https://doi.org/10.5665/sleep.2106.

Vasconcelos, Marco. 2008. “Transitive Inference in Non-Human Animals: An Empirical and Theoretical Analysis.” *Behavioural Processes* 78 (3): 313–34.

Volgushev, Maxim, Jen-Yung Chen, Vladimir Ilin, Roman Goz, Marina Chistiakova, and Maxim Bazhenov. 2016. “Partial Breakdown of Input Specificity of STDP at Individual Synapses Promotes New Learning.” *The Journal of Neuroscience* 36 (34): 8842–55. https://doi.org/10.1523/JNEUROSCI.0552-16.2016.

Wagner, Ullrich, Steffen Gais, Hilde Haider, Rolf Verleger, and Jan Born. 2004. “Sleep Inspires Insight.” *Nature* 427 (6972): 352.

Walker, Matthew P., Tiffany Brakefield, Joshua Seidman, Alexandra Morgan, J. Allan Hobson, and Robert Stickgold. 2003. “Sleep and the Time Course of Motor Skill Learning.” *Learning & Memory* 10 (4): 275–84. https://doi.org/10.1101/lm.58503.

Walker, Matthew P., Conor Liston, J. Allan Hobson, and Robert Stickgold. 2002. “Cognitive Flexibility across the Sleep–Wake Cycle: REM-Sleep Enhancement of Anagram Problem Solving.” *Cognitive Brain Research* 14 (3): 317–24.

Walker, Matthew P., and Robert Stickgold. 2004. “Sleep-Dependent Learning and Memory Consolidation.” *Neuron* 44 (1): 121–33. https://doi.org/10.1016/j.neuron.2004.08.031.

Wei, Yina, Giri P. Krishnan, and Maxim Bazhenov. 2016. “Synaptic Mechanisms of Memory Consolidation during Sleep Slow Oscillations.” *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 36 (15): 4231–47. https://doi.org/10.1523/JNEUROSCI.3648-15.2016.

Wei, Yina, Giri P. Krishnan, Maxim Komarov, and Maxim Bazhenov. 2018. “Differential Roles of Sleep Spindles and Sleep Slow Oscillations in Memory Consolidation.” *PLOS Computational Biology* 14 (7): e1006322. https://doi.org/10.1371/journal.pcbi.1006322.

Wendelken, Carter, and Silvia A. Bunge. 2010. “Transitive Inference: Distinct Contributions of Rostrolateral Prefrontal Cortex and the Hippocampus.” *Journal of Cognitive Neuroscience* 22 (5): 837–47.

Werchan, Denise M., and Rebecca L. Gómez. 2013. “Generalizing Memories over Time: Sleep and Reinforcement Facilitate Transitive Inference.” *Neurobiology of Learning and Memory* 100: 70–76.

Yang, Changkook, and John W. Winkelman. 2006. “Clinical Significance of Sleep EEG Abnormalities in Chronic Schizophrenia.” *Schizophrenia Research* 82 (2): 251–60. https://doi.org/10.1016/j.schres.2005.10.021.

Zalesak, Martin, and Stephan Heckers. 2009. “The Role of the Hippocampus in Transitive Inference.” *Psychiatry Research: Neuroimaging* 172 (1): 24–30.

2 Simulated sleep helps to generalize knowledge in a spiking network trained with spike-timing dependent plasticity

2.1 Abstract

Artificial neural networks are known to generalize poorly to new examples; although they excel at representing data observed in the training set, they are unable to represent data drawn from different distributions. In the mammalian brain, evidence suggests that sleep promotes generalization of learned examples. To address the validity of this hypothesis, we utilized a previously developed spiking neural network trained with spike-timing dependent plasticity (STDP) to perform digit classification on the MNIST dataset. We demonstrate that incorporating an offline, sleep-like period after training leads to generalization and robustness to novel inputs.

2.2 Introduction

Although artificial neural networks (ANNs) can rival human performance on various tasks, ranging from complex games (Silver et al. 2016) to image classification (Krizhevsky, Sutskever, and Hinton 2012), they have been shown to underperform when the testing data differs in specific ways even by a small amount from the training data (Geirhos et al. 2018). This lack of generalization presents several issues when ANNs are utilized in the real world. Primarily, ANNs are often trained on refined datasets of images designed to best capture the image content, whereas in real-world scenarios, they may be tested on disturbed or noisy inputs, not observed during training. Creating more robust neural networks will pave the way forward for using these promising neuro-inspired architectures in the real-world.

It has been hypothesized that in the mammalian brain sleep helps to create generalized representations of the information learned during the awake state (Stickgold 2013). Sleep has been identified as being critical for memory consolidation - a process of converting recent memories

into long-tern storage (Rasch and Born 2013). During sleep, there is reactivation of neurons involved in previously learned activity (Stickgold 2005b) and this reactivation is likely to invoke the same spatio-temporal pattern as the pattern observed during training in the awake state (Wilson and McNaughton 1994). Sleep reactivation, or replay, serves to strengthen synapses involved in a learned task through spike-timing dependent plasticity rules (STDP). Sleep, through STDP, can increase a subject's ability to form logical connections between memories and to generalize knowledge learned during the awake state (Payne et al. 2009).

Similarly, research suggests that sleep can help extract the gist of a task by strengthening connections pertinent to all memories while weakening connections, through synaptic downscaling, relevant to a single, spurious memory (Lewis and Durrant 2011). This body of neuroscience work suggests that a sleep-like phase applied in training neural networks may allow for gist extraction of the training data, leading to increased generalization and robustness to the underlying distribution of the training data. Our hypothesis is that sleep could aid in increasing a neural network's generalization performance by reducing the impact that small additions of noise can have on the network's classification accuracy.

2.3 Methods

2.3.1 Network Architecture and Simulated Sleep

To address the validity of this hypothesis, we utilized a spiking neural network trained with STDP previously proposed to perform digit classification on the MNIST dataset (see (P. Diehl and Cook 2015) for details). The MNIST dataset represents a simple task for artificial intelligence whereby the network must learn to classify grayscale images of handwritten digits (Lecun et al. 1998). The spiking network consists of 3 layers: an input layer, an excitatory middle layer and an inhibitory layer. Neurons in the input layer receive input proportional to the intensity of each pixel

in the MNIST images. The input layer projects to a layer of excitatory neurons with an all-to-all connectivity matrix and the weights of these connections are updated by an STDP rule. In addition, the excitatory layer projects to and receives lateral inhibition (which promotes competition amongst neurons) from the inhibitory layer. The neurons within each layer are governed by leaky-integrate-and-fire dynamics. Additionally, each neuron in the excitatory layer has a threshold parameter which is governed by a homeostatic rule to ensure balanced activity (see Figure 2.1 for a summary of the architecture).

As the network is presented with more images, the network is able to classify a greater percentage of images correctly, by modifying weights from the input to the excitatory layer to compute 2-dimensional spatial filters of the MNIST digits (see Figure 2.2A-C). 2-D receptive fields are computed by reshaping the weights connecting to a single neuron in the excitatory layer into the same dimension as the input images. Then, these 2-D receptive fields are aligned in order to visualize all receptive fields learned by the network.

While this network can learn the task, it only reaches high levels of performance (>80%) after training on more than 100,000 images. We took a partially trained network (between 20 and 80% of the full training image set) and applied a sleep-like phase after the learning phase. During simulated sleep, we modified the intrinsic and synaptic currents to mimic changes in neuromodulator levels, while presenting noisy Poisson input based on the statistics of the MNIST input (see Figure 2.1 for dynamical equation updates). These changes capture cellular and synaptic changes which occur during stage 3 sleep, and result in an increase in activity, mirroring the "up-state" of slow-wave sleep (Wei, Krishnan, and Bazhenov 2016). During sleep, the same STDP and threshold updating rules are used. We compared performance before (awake) and after sleep by computing the classification of the network on different testing images. Classification is done by

assigning each neuron in the excitatory layer a label (0-9) based on which set of digits produce the maximum mean firing activity in that neuron. Networks are tested from various random initializations (n=5) to measure variability of the training and sleep phases.

2.4 Results

2.4.1 Sleep improves performance on networks trained with small dataset

After training in the awake state, the network is able to accurately classify the MNIST digits. However, at different levels of awake training (measured by how many images in the training set the network has observed), incorporating an off-line sleep period after awake training notably increases classification accuracy on a novel test set (Figure 2.3) Most notably, at very small levels of training (1000 images), the trained network classifies the test set with 20% accuracy. However, after a sleep-like period where noisy input is presented to the network, classification accuracy reaches 60%. This effect is pronounced even at higher levels of awake training, suggesting that a sleep-like period can promote one-shot learning and greater generalization of the task structure.

2.4.2 Sleep promotes increased generalization

As noted above, neural network-based classifiers often suffer from poor robustness. If a network is trained on intact, undistorted images, then the network will fail to classify distorted images, even if the distortions are not significant enough to affect human-level perception. To test the effect of sleep on a network's robustness, we added noise to the MNIST images, either by adding random Gaussian noise or applying a blur filter to the images (Figure 2.4A). We found that the network after undergoing a sleep period is able to classify more images correctly even as the images are further distorted (Figure 2.4B). These results mirror the results from biology which

suggest that sleep can help a subject extract the gist of a task and generalize knowledge learned during a waking period.

2.4.3 Sleep decorrelates the representation of distinct digits

We next wanted to ask how sleep is able to give rise to one-shot learning and increased robustness. In biological networks, it has been hypothesized that sleep can generalize knowledge by down-scaling the activity of irrelevant synapses or neurons and strengthening the impact of highly salient neurons (Lewis and Durrant 2011). Additionally, biophysical modelling suggests that sleep can decorrelate the representation of a certain task by devoting synapses to specific memory traces (González et al. 2020). In this network, we observed the same effect, where the representation in the excitatory layer of distinct digits is further decorrelated after sleep as compared to before while the representation of digits from the same class remains highly correlated (see Fig. 2.5).

2.4.4 Sleep prunes task-irrelevant neurons from the network

We next analyzed which component of the network, changing neuronal thresholds or synaptic plasticity, contributed the most to the accuracy increase after sleep. We observed that most neurons experienced an increase in their thresholds due to the constant activity presented during sleep and the homeostatic rule used to change thresholds (Figure 2.6A). However, neurons with well-formed 2-D receptive fields were qualitatively more likely to have decreasing thresholds after sleep (Figure 2.6C). Oppositely, neurons with noisy 2-D receptive fields were more likely to experience an increase in their firing thresholds following sleep (Figure 2.6D). We quantified this phenomenon by looking at the average neighborhood pixel variance using 3x3 pixel squares. Receptive fields with low neighborhood pixel variance are likely to be more refined since there is little variability between neighboring pixels. In contrast, noisy receptive fields should have high

neighborhood pixel variance. There was a significant correlation between neighborhood pixel variance and threshold change (Figure 2.6B), suggesting the hypothesis that sleep improves performance and robustness by pruning task-irrelevant neurons from the network by increasing their firing thresholds.

We confirmed this as the main source of improvement after sleep by testing the network in four conditions: using either before- or after-sleep weights and before- or after-sleep thresholds. The largest performance increase was observed when after-sleep thresholds were used (no significant difference between normal sleep and only using after-sleep thresholds, $p=0.22$). However, when pre-sleep thresholds were used along with the STDP changes that resulted from sleep, performance did not improve significantly. This suggests that in the default network architecture, sleep improves performance by altering the thresholds in a manner in which task-specific neurons can respond more acutely (because of reduced thresholds) to the images presented during testing.

Finally, we analyzed the effect of reducing inhibition and fixing the thresholds during sleep in order to determine the role of synaptic plasticity changes that occur during sleep on generalization. We were able to see the same performance increase after sleep by reducing inhibition in the network, as competition between neurons was reduced (normal sleep vs. only STDP changes, $p=0.85$). We explored the synaptic weight changes during sleep and uncovered two main principles (Fig 2.6). First, in neurons with well-formed receptive fields, there is very little synaptic weight change after sleep. Second, in neurons with task-irrelevant receptive fields, there is an overall synaptic down-scaling of connections, mirroring the results from the threshold analysis above. Overall, these results support the role of sleep in memory consolidation and generalization of knowledge learned during the waking state. Moreover, this line of work supports

the synaptic homeostasis hypothesis of sleep which suggests that slow-wave sleep improves performance by down-scaling synaptic weights (Tononi and Cirelli 2006).

2.5 Discussion

In this study, we applied an off-like sleep-like phase to the training phase of a spiking network trained to perform the MNIST digit classification task. We found that after any amount of awake training, adding a sleep phase, where noisy Poisson input is passed through the network and activity is elevated, can increase the classification accuracy on a novel test set. Similarly, the network after sleep is able to respond to more diverse representations of the image set, classifying noisy and blurred images more accurately than before sleep. These results mirror work in biology which has shown that sleep can help extract the gist of a task and generalize knowledge learned during the awake state (Stickgold 2013). Additionally, these results lend support to the synaptic homeostasis hypothesis which suggests that sleep down-scales synaptic weights to make efficient use of brain space in an energy-conserving manner (Tononi and Cirelli 2006). Our experiments suggests that down-scaling of synaptic activity is likely constrained to task-irrelevant neurons, thereby containing the representation of the task to a subset of neurons.

For artificial intelligence systems, these results suggest that the same classification accuracy can be achieved by adding a sleep phase after either a shorter length awake-training period or with a reduced dataset, giving rise to one-shot learning. Similarly, incorporating a sleep-phase to the training of artificial neural networks can increase the network's robustness to noisy images or imperfect pictures captured in the real-world.

2.6 Acknowledgements

Chapter 2, in full, is a reprint of the material as it was published at the Bridging AI and Cognitive Science Workshop at ICLR 2020. Tadros, Timothy; Krishnan, Giri; Bazhenov, Maxim. BAICs Workshop, 2020. The dissertation author was one of the primary investigators and first author on this paper.

2.7 Figures

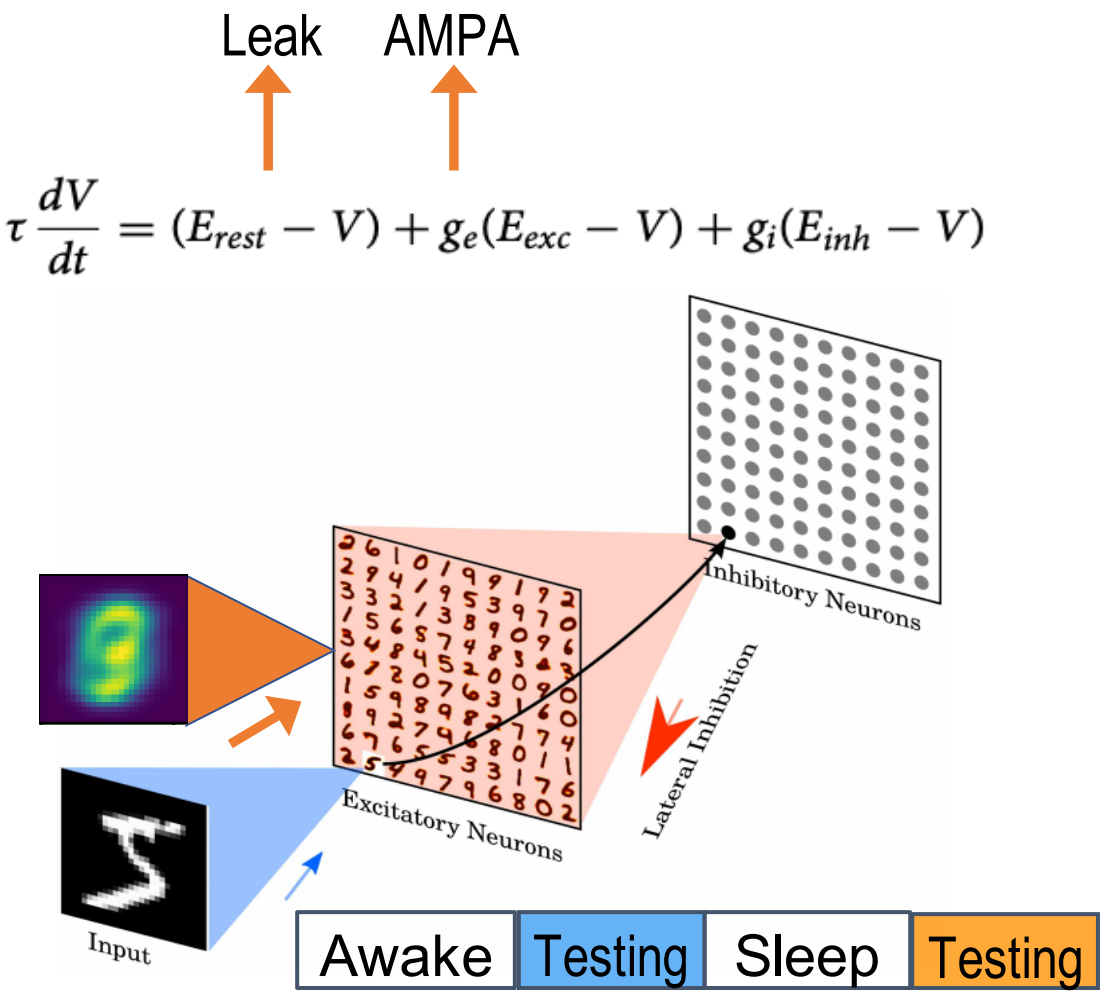

Figure 2.1: Network architecture and sleep changes

(A) Network schematic showing the basic network architecture (adapted from (Diehl and Cook 2015)).
(B) Changes to network applied during sleep include presenting the average image and increasing leak and AMPA currents.

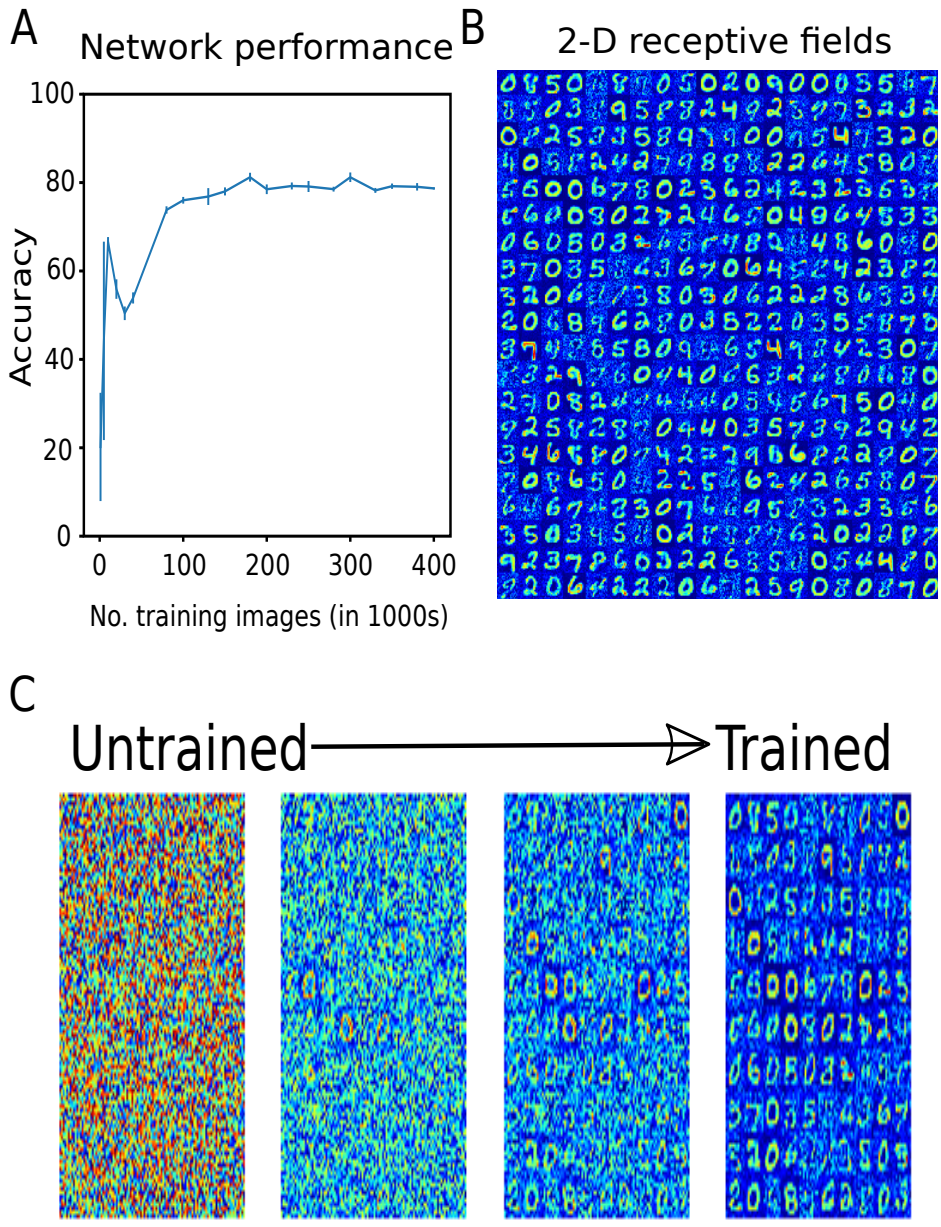

Figure 2.2: Network is able to learn the digit classification task

(A) Accuracy as a function of number of images seen.
(B) Receptive fields of neurons in the excitatory layer form into 2-d spatial filters. Network receptive fields at different stages in baseline (before sleep) training reveal that traininh refines the receptive fields of excitatory layer neurons.

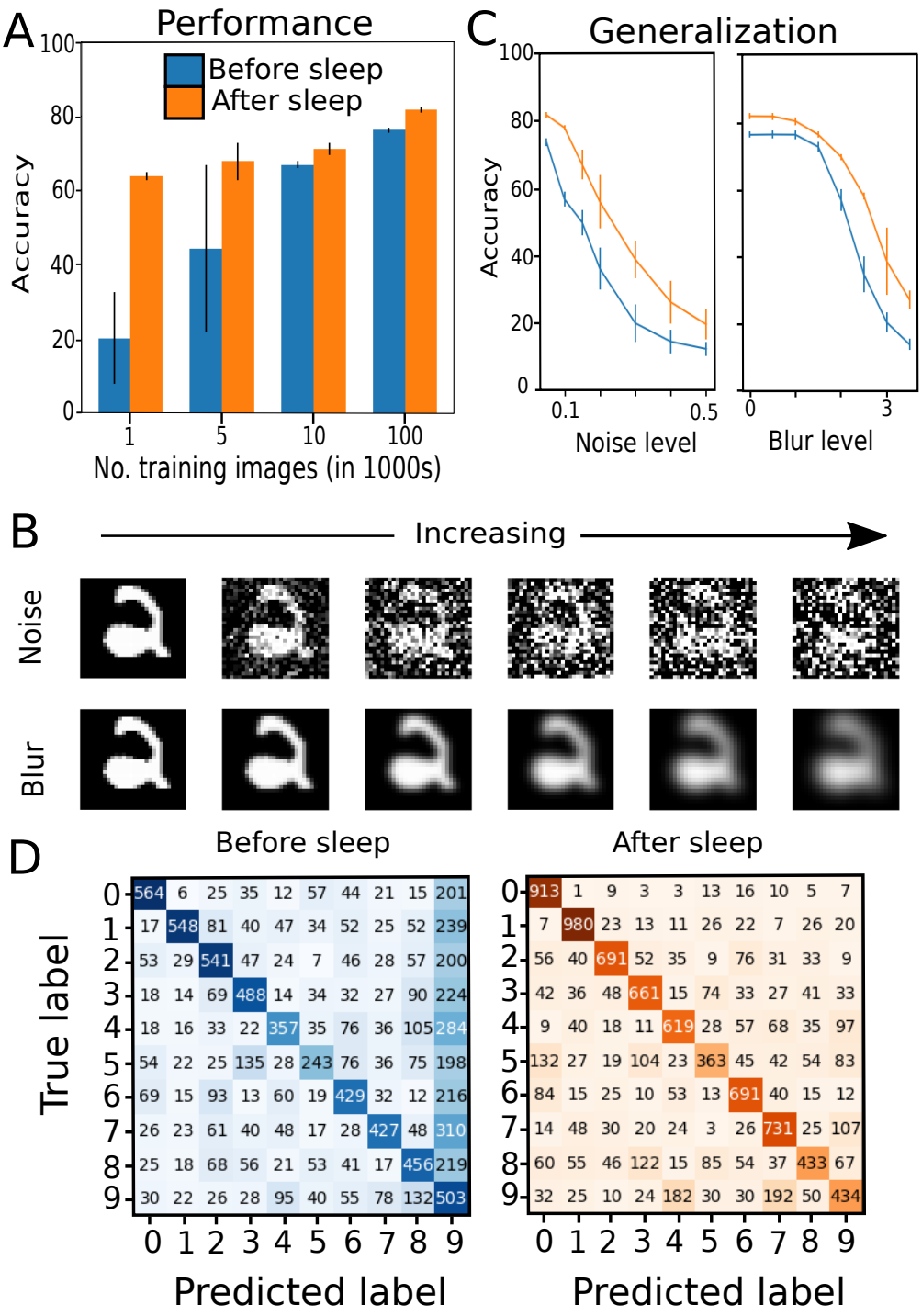

Figure 2.3: Sleep improves accuracy and generalization

(A) Accuracy before and after sleep for different training levels.
(B) Example images for increasing levels of noise and blur.
(C) Accuracy before and after sleep on various noise conditions (10,000 image network).
(D) Confusion matrices on undistoreted images before and after sleep.

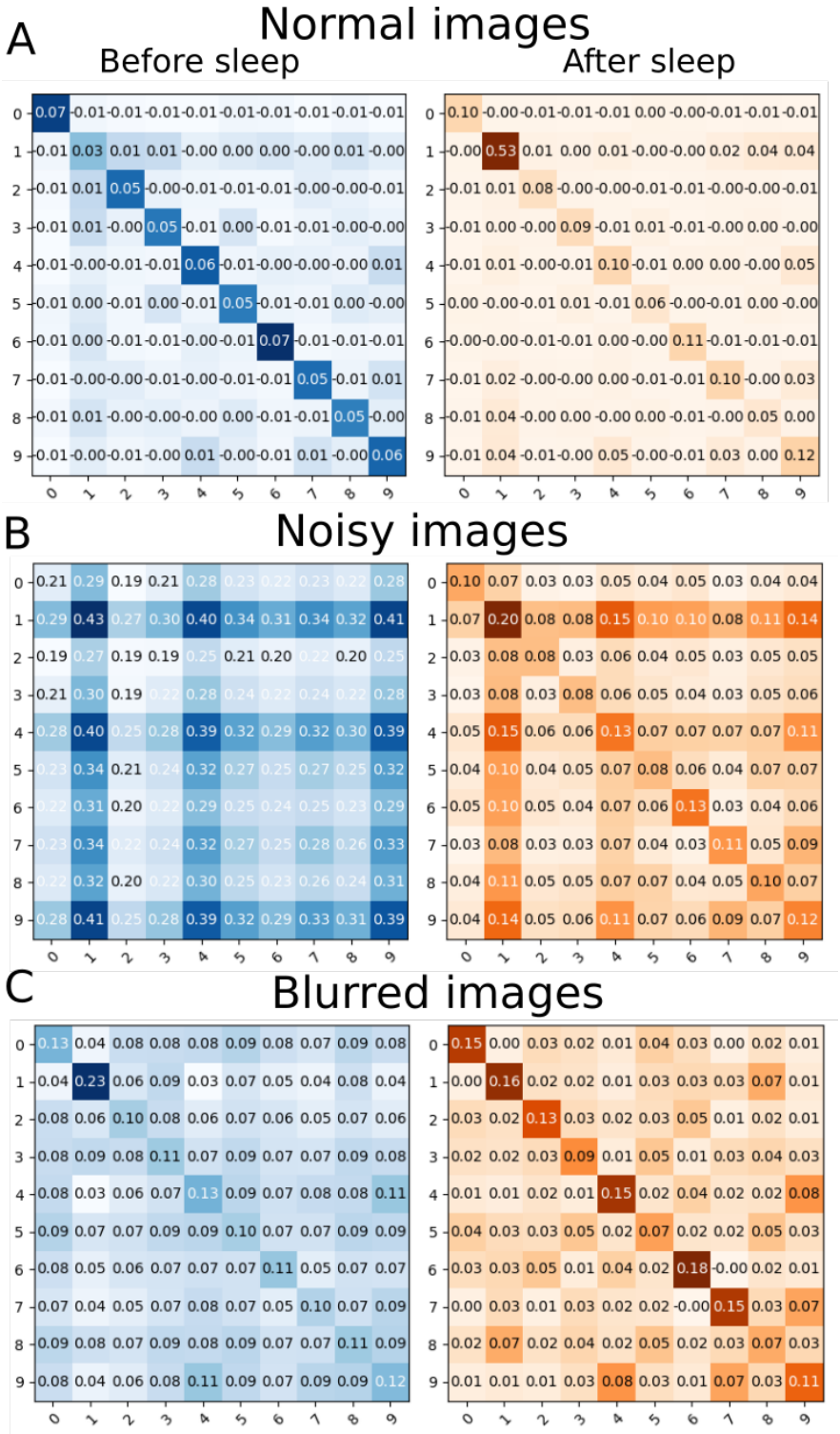

Figure 2.4: Decorrelated representation of distinct digits after sleep

(A) Pairwise correlations for normal images before and after sleep. Correlations computed in the excitatory layer of the network.
(B) Noisy images.
(C) Blurred images.

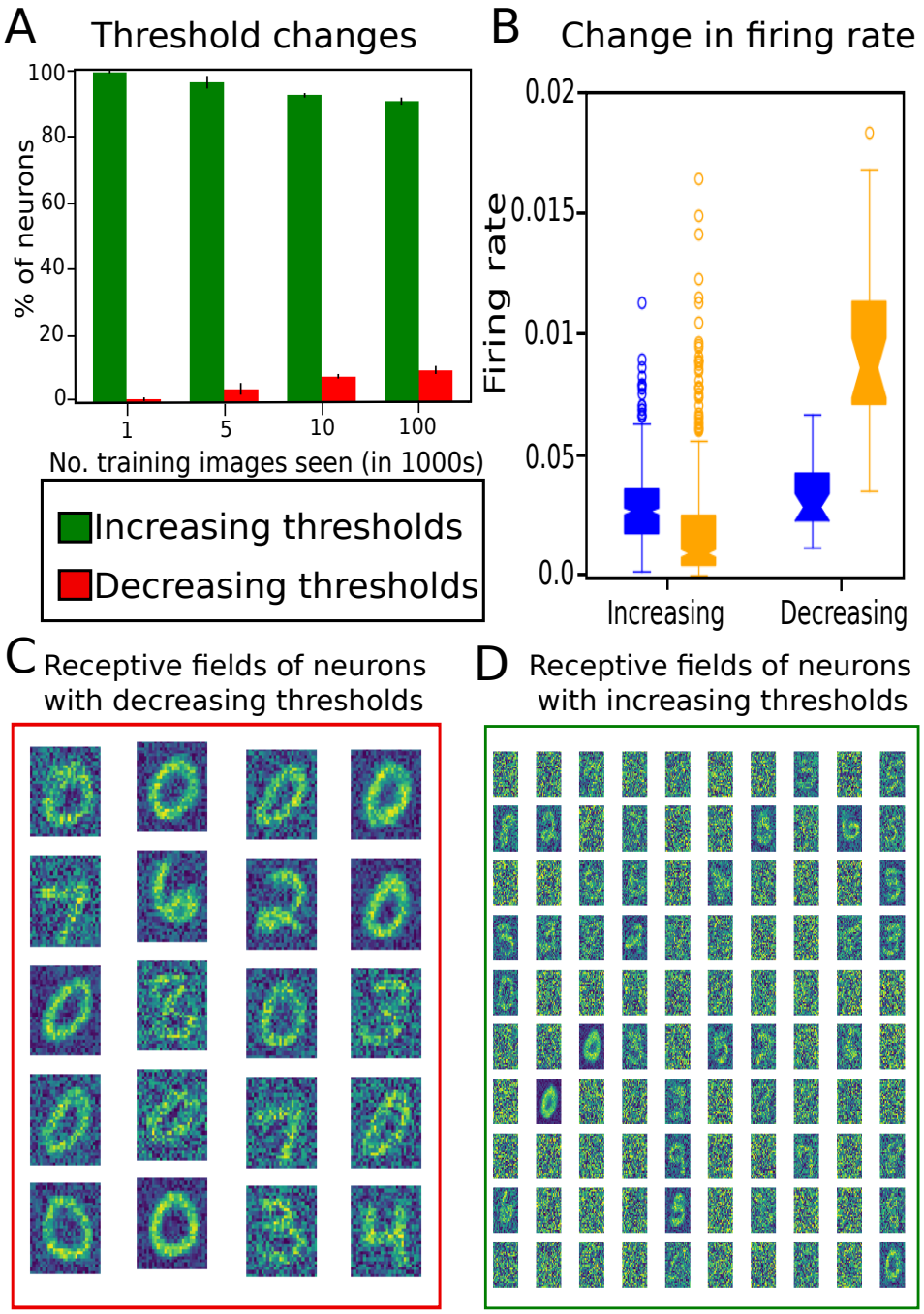

Figure 2.5: Task-irrelevant neurons fire less after sleep

(A) Number of neurons that have increasing or decreasing thresholds after sleep at various stages of training.
(B) Firing rates of neurons before and after sleep for neurons with increasing or decreasing thresholds.
(C) Example receptive fields for neurons with decreasing thresholds.
(D) Example receptive fields for neurons with increasing thresholds.

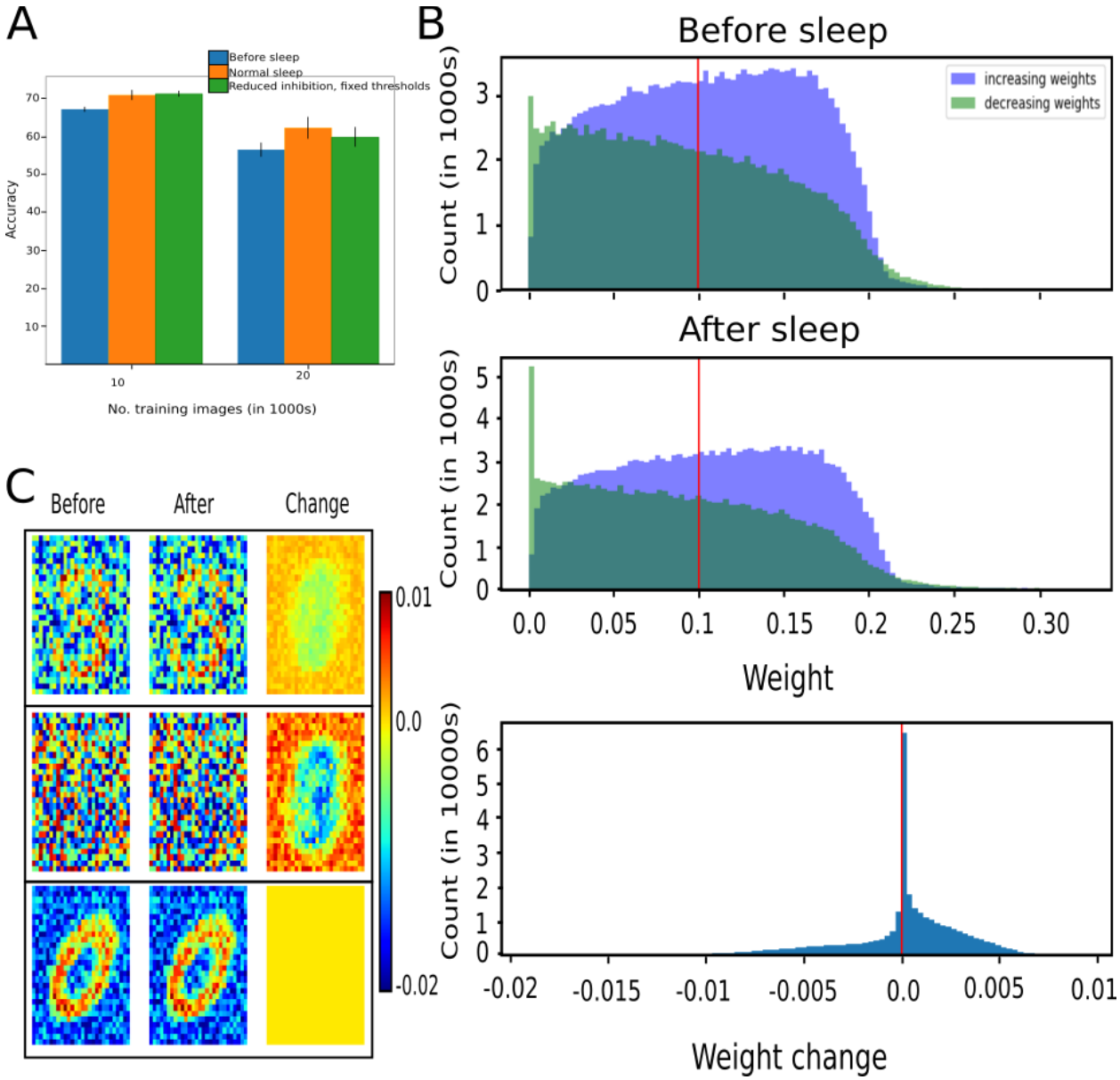

Figure 2.6: STDP alone can achieve the same effect by reducing inhibition and fixing thresholds during sleep

(A) Accuracy before sleep, after normal sleep, and after a sleep period with only STDP.
(B) Weight distributions before (top) and after-STDP sleep (middle).
(C) Example canonical receptive field changes – noisy receptive fields see a decrease in weight magnitude in the center, while well-formed receptive fields see little change.

2.8 References

Diehl, Peter, and Matthew Cook. 2015. “Unsupervised Learning of Digit Recognition Using Spike-Timing-Dependent Plasticity.” *Frontiers in Computational Neuroscience* 9. https://www.frontiersin.org/article/10.3389/fncom.2015.00099.

Geirhos, Robert, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. 2018. “Generalisation in Humans and Deep Neural Networks.” In *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2018/hash/0937fb5864ed06ffb59ae5f9b5ed67a9-Abstract.html.

González, Oscar C, Yury Sokolov, Giri P Krishnan, Jean Erik Delanois, and Maxim Bazhenov. 2020. “Can Sleep Protect Memories from Catastrophic Forgetting?” Edited by Mark CW van Rossum, Laura L Colgin, and Francesco P Battaglia. *ELife* 9 (August): e51005. https://doi.org/10.7554/eLife.51005.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. “ImageNet Classification with Deep Convolutional Neural Networks.” In *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner. 1998. “Gradient-Based Learning Applied to Document Recognition.” *Proceedings of the IEEE* 86 (11): 2278–2324. https://doi.org/10.1109/5.726791.

Lewis, Penelope A., and Simon J. Durrant. 2011. “Overlapping Memory Replay during Sleep Builds Cognitive Schemata.” *Trends in Cognitive Sciences* 15 (8): 343–51. https://doi.org/10.1016/j.tics.2011.06.004.

Payne, Jessica D., Daniel L. Schacter, Ruth E. Propper, Li-Wen Huang, Erin J. Wamsley, Matthew A. Tucker, Matthew P. Walker, and Robert Stickgold. 2009. “The Role of Sleep in False Memory Formation.” *Neurobiology of Learning and Memory* 92 (3): 327–34. https://doi.org/10.1016/j.nlm.2009.03.007.

Rasch, Björn, and Jan Born. 2013. “About Sleep’s Role in Memory.” *Physiological Reviews* 93 (2): 681–766. https://doi.org/10.1152/physrev.00032.2012.

Silver, David, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, et al. 2016. “Mastering the Game of Go with Deep Neural Networks and Tree Search.” *Nature* 529 (7587): 484–89. https://doi.org/10.1038/nature16961.

Stickgold, Robert. 2005. “Sleep-Dependent Memory Consolidation.” *Nature* 437 (7063): 1272–78. https://doi.org/10.1038/nature04286.

Stickgold, Robert, and Matthew P. Walker. 2013. “Sleep-Dependent Memory Triage: Evolving Generalization through Selective Processing.” *Nature Neuroscience* 16 (2): 139.

Tononi, Giulio, and Chiara Cirelli. 2006. “Sleep Function and Synaptic Homeostasis.” *Sleep Medicine Reviews* 10 (1): 49–62. https://doi.org/10.1016/j.smrv.2005.05.002.

Wei, Yina, Giri P. Krishnan, and Maxim Bazhenov. 2016a. “Synaptic Mechanisms of Memory Consolidation during Sleep Slow Oscillations.” *The Journal of Neuroscience* 36 (15): 4231. https://doi.org/10.1523/JNEUROSCI.3648-15.2016.

Wilson, M. A., and B. L. McNaughton. 1994. “Reactivation of Hippocampal Ensemble Memories during Sleep.” *Science* 265 (5172): 676–79. https://doi.org/10.1126/science.8036517.

3 Biologically Inspired Sleep Algorithm for Increased Generalization and Adversarial Robustness in Deep Neural Networks

3.1 Abstract

Current artificial neural networks (ANNs) can perform and excel at a variety of tasks ranging from image classification to spam detection through training on large datasets of labeled data. While the trained network may perform well on similar testing data, inputs that differ even slightly from the training data may trigger unpredictable behavior. Due to this limitation, it is possible to design inputs with very small perturbations that can result in misclassification. These adversarial attacks present a security risk to deployed ANNs and indicate a divergence between how ANNs and humans perform classification. Humans are robust at behaving in the presence of noise and are capable of correctly classifying objects that are noisy, blurred, or otherwise distorted. It has been hypothesized that sleep promotes generalization of knowledge and improves robustness against noise in animals and humans. In this work, we utilize a biologically inspired sleep phase in ANNs and demonstrate the benefit of sleep on defending against adversarial attacks as well as in increasing ANN classification robustness. We compare the sleep algorithm's performance on various robustness tasks with two previously proposed adversarial defenses - defensive distillation and fine-tuning. We report an increase in robustness after sleep phase to adversarial attacks as well as to general image distortions for three datasets: MNIST, CUB200, and a toy dataset. Overall, these results demonstrate the potential for biologically inspired solutions to solve existing problems in ANNs and guide the development of more robust, human-like ANNs.

3.2 Introduction

Although artificial neural networks (ANNs) have recently begun to rival human performance on various tasks, ranging from complex games (Silver et al. 2016) to image

classification (Krizhevsky, Sutskever, and Hinton 2012), ANNs have been shown to underperform when the testing data differs in specific ways even by a small amount from the training data (Geirhos et al. 2018). This lack of generalization presents two issues when ANNs are utilized in the real world. First, ANNs are often trained on curated datasets of images designed to best capture the image content, whereas in real-world scenarios, they may be tested on disturbed or noisy inputs, not observed during training. Second, ANNs are susceptible to adversarial attacks, or the deliberate creation of inputs designed to fool ANNs that may be imperceptibly different from correctly classified inputs (Szegedy et al. 2013). These two issues limit ANNs applicability in the real world and present potential security risks when deployed.

There have been two main approaches for investigating ANN robustness: adversarial machine learning and training data manipulation (Ford et al. 2019). Adversarial machine learning aims to develop novel attack methods which perturb the input minimally while changing the ANN's classification outcome (Moosavi-Dezfooli, Fawzi, and Frossard 2016; Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2014; Athalye et al. 2017; Nguyen, Yosinski, and Clune 2015) as well as to design defense mechanisms which prevent these attacks from affecting ANN behavior (Papernot, McDaniel, Wu, et al. 2016; Goodfellow, Shlens, and Szegedy 2014; Huang et al. 2015). see (Yuan et al. 2019) for review). Training data manipulation research typically examines the impact of changing the input distribution during testing and observing the effect on ANN performance. (Geirhos et al. 2018) showed that ANNs trained on images with one type of distortion may not perform well when tested on other types of distortions, even if images with both distortions appear identical to the human eye. Likewise, ANNs trained on unperturbed images exhibit reduced performance when images in the test set are distorted, for example, through horizontal translations, blurring, or the addition of compression artifacts (Dodge and Karam 2016;

Vasiljevic, Chakrabarti, and Shakhnarovich 2016; Zhou, Song, and Cheung 2017).Although it has been proposed that adversarial and manipulation robustness can be increased through various mechanisms during the training phase, such as fine-tuning, recent research has shown that these methods are mostly ineffective or their effectiveness is inconclusive (Geirhos et al. 2018; Uesato et al. 2018; Athalye, Carlini, and Wagner 2018).

It has been hypothesized that in the mammalian brain sleep helps to create generalized representations of an input learned during the awake state (Stickgold 2013; Lewis and Durrant 2011). Sleep has been identified as being critical for memory consolidation - a process of converting recent memories into long-term storage (Rasch and Born 2013).. During sleep, there is reactivation of neurons involved in previously learned activity (Stickgold 2005a) and this reactivation is likely to invoke the same spatio-temporal pattern of neuronal firing as the pattern observed during training in the awake state (Wilson and McNaughton 1994). Sleep reactivation, or replay, serves to strengthen synapses involved in a learned task through local synaptic plasticity, such as Spike Time Dependent Plasticity (STDP). Plastic changes during sleep can increase a subject's ability to form connections between memories and to generalize knowledge learned during the awake state (Payne et al. 2009). In one study (Wamsley et al. 2010), subjects learned to find an exit to a maze in a virtual 3D environment. Subjects who were allowed to sleep exhibited a more complex understanding of the overall shape of the maze (Wamsley et al. 2010). Using biophysical model of a cortical network (Gonzalez et al. 2019; Wei et al. 2018) showed that sleep dynamics promotes reactivation and helps to create distinct representations for unique memories by devoting synapses to specific memory traces. This body of neuroscience work suggests that a sleep-like activity may be applied to ANNs to enable the network to extract the gist of the training data without being constrained by the statistics of a specific training data set. Our specific

hypothesis is that sleep phase could aid in reducing a neural network's susceptibility to adversarial attacks and to increase generalization performance by reducing the impact that imperceptible input changes can have on the task output.

In this new work, we propose a sleep-inspired algorithm to defend against adversarial attacks as well as to increase ANN robustness to noise. We utilize the notion of sleep from biology and apply an off-line unsupervised "sleep" phase to modify the parameters of a fully connected ANN. We demonstrate a number of performance improvements over existing defense algorithms, such as fine-tuning or adversarial retraining and defensive distillation, on both adversarial and noise robustness. The contributions are summarized below:

- We analyze how robust the proposed sleep algorithm is to four different types of adversarial attacks on three different datasets (MNIST, CUB200, and a toy dataset). For most conditions (MNIST, toy dataset), after sleep phase was applied, the attacks consistently resulted in adversarial examples that were more distinct from the original input compared to the adversarial examples designed for the original (before sleep) network.
- We illustrate that the sleep algorithm creates a more robust network whereby performance on noisy and blurred inputs is higher compared to control or defensively distilled network and is more robust to the other types of distortions compared to ANNs that are fine-tuned on a single distortion.
- We analyze the impact of the sleep algorithm on task representation and demonstrate that the algorithm creates decision boundaries that more closely resemble the true classes, effectively extracting the gist of the data.

3.3 Adversarial Attacks and Distortions

Adversarial attacks aim to create minimal perturbations that, while imperceptible to the human eye, fool ANNs. These attacks range from white-box to black-box attacks, based on how much information they assume the attacker to possess about the network. White-box attacks assume that the attacker has access to the network architecture, training data and weights. These attacks can range from absolute information, such as gradient-based attacks which compute the gradient of the loss with respect to the input (Brendel, Rauber, and Bethge 2017), to score-based attacks which only utilize predicted scores of the model. Black-box attacks, which assume no knowledge about the network, solely rely on the decision made in order to craft adversarial examples. Attacks can be (a) targeted such that the attacker aims to create an adversarial example that the network would predict as a certain class or (b) untargeted where the attacker's goal is simply to cause any kind of misclassification (Biggio and Roli 2018). In this work we consider four types of adversarial attacks ranging from white-box to black-box attacks. We assume that the attacker solely wants to cause a misclassification, with no respect to the output class. We present a brief description of each of the four attacks below (see Figures for examples of images created by these attacks).

3.3.1 Fast Gradient Sign Method (FGSM)

FGSM (Goodfellow, Shlens, and Szegedy 2014) computes the sign of the gradient of the loss function (J) with respect to the original input x using the weights θ of the network and the target labels y.

$$x' = x + \varepsilon sign(\nabla_x J(\theta, x, y))$$

This represents the direction to change each pixel in the original input in order to increase the loss function. Based on the value of ε, the corresponding perturbation to the original image can range

from small to large. Thus, in this work we use the average of the smallest values of ε needed to create an adversarial example x' (misclassified input) for each input in the testing set.

3.3.2 DeepFool

DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016) is an iterative method which approximates the nearest decision boundary to the input at time t and moves the input x_t in that direction to compute x_{t+1}. This process is repeated until a misclassification is produced or the runtime of the simulation is exceeded. For this attack, we measure the L2-norm between the original input x and the adversarial input x'. Thus, successful defenses should result in a high L2-norm for this algorithm.

3.3.3 Jacobian-based Saliency Map (JSMA)

JSMA (Papernot, McDaniel, Jha, et al. 2016) aims to craft adversarial examples that minimize the L0-norm of x – x' by reducing the number of pixels that are altered. In summary, the algorithm computes the gradient, as done in FGSM but for all possible classes. These gradient values represent how changing each pixel contributes to the overall loss function, with large values indicating a significant effect on the loss. These values are used to create a saliency map, where each pixel's impact on the loss is modelled. The algorithm utilizes this saliency map to alter individual pixels, repeating the gradient and saliency map computation until an adversarial example is created. For this type of attack, we utilize the L2-norm to determine defense success.

3.3.4 Boundary Attack

The Boundary Attack (Brendel, Rauber, and Bethge 2017) is a black-box attack which relies solely on the decision of the ANN to craft an adversarial example. Given an input x, a random input x'_0 is chosen such that f(x) != f(x'_0), where f(x) is the label produced by the ANN. In our work, x'_0 is chosen from a uniform distribution. The attack starts by moving x'_0 toward x until it

reaches the point where f(x) = f(x'$_0$), or the decision boundary in between f(x) and f(x'$_0$). From here, the attack consists of two steps: an orthogonal perturbation and a forward perturbation. During the orthogonal perturbation, random points along the hypersphere around f(x'$_t$) are sampled. Those that are adversarial and closer to x than before are added to the queue for forward perturbation. During the forward perturbation, a small step is taken from x'$_t$ to x as long as f(x) != f(x'$_t$). This process is repeated until a convergence criterion is met. For this attack, we utilize the L2-norm to define defense success.

3.3.5 Distortions

Although not specifically designed to attack an ANN, distortions negatively impact ANN performance. In this work we consider two simple distortion techniques: blurring and Gaussian noise. In first case, we perform 2-D Gaussian filtering with a blur kernel of varying standard deviation in order to blur the images. In second case, we add Gaussian noise with mean 0 and standard deviation σ. These distortions are tested in the networks implementing the proposed sleep algorithm as well as using the adversarial defenses discussed below.

3.4 Adversarial Defenses

We compare our sleep algorithm with two existing adversarial defenses: defensive distillation and fine-tuning, or adversarial retraining. Defensive distillation (Papernot, McDaniel, Wu, et al. 2016) utilizes two training sessions in order to create a distilled network. First, an initial network is trained on (X,Y), where X is the training data, and Y is the one-hot encoded training labels. The activation function of this network is changed such that the softmax function of the output layer is computed using a temperature term T as follows:

$$F(x) = \frac{e^{\frac{z_i X}{T}}}{\sum_{l=0}^{N-1} e^{z_l X/T}}$$

A higher T forces the ANN to produce larger probability values for each class, whereas lower T values support a similar representation as the one-hot encoded labels. After the first network is trained, the output of the network (probability values) is used to train a distilled network with the same softmax-temperature function. Previous work has shown this approach can be successful at preventing some types of attacks (Papernot, McDaniel, Wu, et al. 2016). However, others have shown that it is not successful at defending against modified versions of those attacks or novel attacks in general (Carlini and Wagner 2016; 2017). Based on the previous work which found that temperature values between 20 and 100 effectively prevent adversarial attacks (Papernot, McDaniel, Wu, et al. 2016), we use a temperature value of T = 50 in our implementation of defensive distillation.

Adversarial retraining aims to fine-tune the network on adversarial examples with the correct labels as a form of regularization. Previous work has shown that adversarial retraining can mitigate the effectiveness of some adversarial attacks. (Goodfellow, Shlens, and Szegedy 2014) showed that adversarial retraining can reduce the error rate on MNIST, demonstrating greater ANN robustness after fine-tuning. Likewise, (Moosavi-Dezfooli, Fawzi, and Frossard 2016) showed that fine-tuning on DeepFool attacks can reduce the effectiveness of their attacks. However, they observed that fine-tuning on FGSM attacks has negative results, actually increasing the strength of the attack. This suggests that fine-tuning may overfit the network to certain attacks, while failing to extrapolate to other attacks, similar to results shown for generalization in ANNs (Geirhos et al. 2018). For the adversarial retraining procedure presented here, we train the network on the original input and then fine-tune the network on various adversarial attacks with a reduced learning rate.

3.5 Sleep Algorithm

The basic intuition behind the sleep algorithm is that a period of offline activity, whereby network weights are modified according to an unsupervised learning algorithm, allows the parameters of the network to become more reflective of the underlying statistics of the task at hand, while not overfitting the statistics of the training data. The pseudocode is presented in Figure 3.1. In short, an ANN is trained using stochastic gradient descent and the standard backpropagation algorithm (exact parameters used for each of the datasets are shown in Table 3.3). After training, the network structure is converted into a spiking neural network (SNN). After building the SNN, we run a 'sleep' phase which modifies the network connectivity based on spike-timing dependent plasticity (STDP). After the sleep phase, the SNN network is converted back into the ANN and testing is performed.

3.5.1 Spiking Neural Networks

SNNs seek to model closely temporal brain dynamics. In short, SNNs are composed of spiking neurons and model the information transformation and the dependence on exact timing of spikes that occurs in biological networks (Ghosh-Dastidar and Adeli 2009). Individual neuron models can range from simple integrate-and-fire type neurons which sum their inputs and produce an output (spike) if this exceeds some firing threshold to more complex Hodgkin-Huxley type neurons which model sodium-, potassium-, and chloride-channel kinetics (Abbott and Kepler 1990). Recent work has shown that a near loss-less conversion between ANNs and SNNs can be achieved by propagating activity through a spiking neural network for a given input and counting the number of times that each output neuron fires (P. U. Diehl et al. 2015).

To convert an ANN to SNN (Lines 1-3 of pseudocode), we assume the ANN utilizes ReLU neurons with no bias. This assumption is made so that the output neuron's activation can be treated

as a firing rate, either zero or positive, and that the thresholds of all neurons in a given layer are of the same scale. The weights from the ANN are directly mapped to the SNN. In our analysis, each unit in the SNN is modelled as an integrate-and-fire type neuron, computing the following equation:

$$\tau_m \frac{dv}{dt} = -v(t) + \sum_{i=1}^{N} w_i * s(i)$$

Here, τ_m represents the decay constant of the membrane potential, v is the voltage at a given time, w_i is the weight connecting from neuron u, and $s(i)$ is the spiking activity of neuron i, either 1 or 0.

3.5.2 Plasticity and Sleep

The key advantage of using a SNN is that biologically inspired training rules can be applied while the network is driven by noisy input. Empirical data suggest that the brain uses spike-timing dependent plasticity (STDP) (Song, Miller, and Abbott 2000), where weight updates depend on the relative timing of pre- and post-synaptic spikes. It has been shown that STDP results in balanced activity, where all neurons fire in equal proportions (Song, Miller, and Abbott 2000). Here we utilize a modified version of STDP: if a pre-synaptic spike induces a post-synaptic spike, then the weight between these neurons is increased. If a post-synaptic spike occurs, but the pre-synaptic neuron does not spike, then the corresponding weight is decreased (in this case postsynaptic spiking may occur because of spiking in other neurons connecting to that post-synaptic neuron).

The sleep training phase we propose here can be described as following. First, inputs to each neuron of the input layer must be presented as spiking activity in order to propagate activity from the input layer to the hidden layers of the network. We convert inputs (real-valued pixel

intensities or features) to spikes by defining a maximum firing rate f_{max} with units spikes/sec and computing a Poisson-distributed spike raster, such that inputs with higher values (i.e. brighter pixels) will have higher rate than inputs with lower values, with no spike rates exceeding f_{max}. Next, activity is propagated through the network as spikes and the STDP rule is applied to update weights. In biological networks, increase of synaptic strength during slow-wave sleep leads to characteristic patterns of activity with repetitive periods of elevated firing (Up-states), when previously learned memory traces are spontaneously replayed. To simulate this dynamics, synaptic weights in SNN are up-scaled to induce high firing rates in later layers. Other important parameters include the threshold for each layer and the length of sleep. The parameters used for each dataset are presented in Table 3.2.

3.5.3 Experiments and Datasets

Below, we describe the general experimental setup as well as the datasets tested. First, we trained a control ANN using the training set for each of the main datasets used in this study. Next, we created a defensively distilled network using T = 50 for the temperature parameter to create the second test network. Then, we fine-tuned the control ANN on a specific attack or distortion method to create the third test network. Finally, we converted the control ANN to an SNN and applied the sleep algorithm as described above to create the fourth test network. We created adversarial examples for each of these four networks using the attacks we described above (fine-tuned networks are tested on the attacks they were fine-tuned on). Then, we analyze how successful each attack is to fool each of the four networks using the metrics defined above. For generalization (blur and noise), we performed the same setup as above creating four different networks. We then tested each network on varying levels of distortion. We tested networks fine-tuned on blurred and noisy

images to measure how performance generalizes across distortion methods. We averaged performance across a minimum of three networks for each attack and distortion.

We used three datasets to compare performance: Patches (a toy dataset created simply for analysis), MNIST (Lecun et al. 1998), and CUB-200 (Welinder et al. 2010). Patches consists of four binary images arranged in a 10x10 square. Each image has its own label (1-4) and consists of 25 bright pixels (value set to 1) and 75 dark pixels. The overlap of bright pixels among the four images (see Appendix) is chosen such that the task is not trivial. The MNIST dataset consists of 70,000 28x28 greyscale images of handwritten digits, with 60,000 in the training set and 10,000 in the testing set. CUB-200 is a high-resolution dataset of images of birds with 200 bird species, with very few (~30) images per class. For this dataset, we used previously extracted ResNet-50 embeddings, where ResNet-50 was pre-trained on ImageNet (He et al. 2016). For CUB-200, we do not report results for blurring, since we are using extracted features, not images.

3.6 Results

We evaluate the sleep algorithm in two settings: (1) Adversarial attacks designed to fool neural networks and (2) generalization distortions designed to reflect imperfect viewing conditions or other types of noise. For adversarial attacks (other than FGSM), we utilize the following metric to evaluate the success of each defense. Let x_i' be the adversarial example created for input x_i. The total score S_A for an attack is the median squared L2-distance for all samples, where N is the dimension of the space:

$$S_A = median(\frac{1}{N} |x'_i - x_i|_2^2)$$

For FGSM, we define the following metric which computes the median minimum noise level ε needed to produce a misclassification across all samples:

$$S_{FGSM} = median(\min(\varepsilon_i)\ s.t.\ f(x_i + \varepsilon_i x'_i) != f(x_i)))$$

For MNIST and CUB-200, we evaluate the attacks on all examples in the testing set. Examples that the networks get wrong before the attack was implemented are discarded from the analysis (in these cases $|x'_i - x_i|_k^2 = 0$ and $\varepsilon_i = 0$ for all attacks). For FGSM and distortions, we also include plots of classification accuracy as a function of noise level. For DeepFool and JSMA, we report adversarial attack accuracy (number of examples where f(x) = y and f(x') ! = f(x), where y is the correct label, over number of examples tested). Note that these algorithms would always produce an adversarial example if allowed to run forever. However, due to computational limitations, we included run-time limits on the number of iterations for these algorithms (see Appendix). Thus, a lower adversarial attack accuracy indicates that the attack would need more iterations to run to reach 100% accuracy. This is a similar measure as distance since more iterations would result in more distinct adversaries for all attacks implemented and the updates at each iteration have the same magnitude for each defense.

3.6.1 Adversarial Attacks

Here we report the scores for all different attacks and for the all datasets. For the FGSM attack, the sleep algorithm increases the median minimum noise needed for misclassification for all three datasets compared to the control network (also see Figure 2). For MNIST dataset, the amount of noise needed to fool the network after the sleep algorithm was almost double of that needed for either the fine-tuning or defensive distillation approaches. For the Patches dataset, both defensive distillation and fine-tuning increase the robustness of the network. However, on CUB-200, only fine-tuning and sleep were able to defend, albeit marginally, against the FGSM attack. Looking at the classification accuracy of the network as a function of noise added (ε, Figure 2), we observe that in the Patches and CUB-200 dataset, sleep tends to have higher classification accuracy than the other methods for epsilon < 0.1. After this point, sleep tends to have equal

classification accuracies as compared to the other methods. For MNIST, the baseline classification accuracy on the original test set decreases slightly compared to the other methods (80% after sleep). However, the performance remains high longer than for the other defense methods on images that were correctly classified. We observed that performance continued to drop after a sufficiently large amount of noise was added. This is biologically plausible as adding more noise to an image should result in image degradation and misclassifications. In sum, these results indicate that a sleep phase can successfully mitigate FGSM, more so than a control network.

Table 3.1: Adversarial Attack Scores (Best defense scores are bolded, lowest attack success rates are in blue).

Patches	**Control**	**Defensive Distillation**	**Fine-tuning**	**Sleep**
FGSM	0.0175	0.05	0.1425	**0.2025**
DeepFool	**0.0440** (95%)	0.036 (90%)	0.0201 (100%)	0.0125 (100%)
JSMA	0.0049 (80%)	0.0135 (70%)	0.0450 (100%)	**0.0541** (100%)
Boundary Attack	0.2971	0.3124	0.1772	**0.3515**
MNIST				
FGSM	0.0900	0.0900	0.1000	**0.2200**
DeepFool	0.0042 (96.46%)	0.0043 (96.42%)	0.0074 (97.40%)	**0.0484** (86.38%)
JSMA	0.0477 (95.56%)	0.0347 (99.41%)	**0.0530** (98.77%)	0.0059 (72.97%)
Boundary Attack	0.0525	0.0525	**0.0544**	0.0488
CUB-200				
FGSM	0.0550	0.0500	**0.0650**	0.0600
DeepFool	0.0027 (82.23%)	0.0019 (84.16%)	**0.0044** (83.02%)	0.0025 (83.12%)
JSMA	0.0477 (95.56%)	0.0347 (95.88%)	**0.0540** (95.38%)	0.0439 (95.15%)
Boundary Attack	0.9751	0.9034	**0.9976**	0.9957

For DeepFool, sleep has a significant effect on the defense score on the MNIST dataset, both reducing the attack success rate and increasing the distance between the adversarial example and the original input by an order of magnitude. For Patches and CUB-200 this effect is less pronounced, with fine-tuning or the control network performing better. We hypothesize that sleep was ineffective in preventing the DeepFool attack in tasks with very few exemplars per class

(Patches) or a large number of classes (CUB-200). In CUB-200, there is a large number of classes so the distance between the input and the nearest decision boundary is smaller (this is supported by the fact that JSMA, an L0 attack, does worse than DeepFool for CUB-200 and vice versa for MNIST, control networks). In this case, sleep is unable to move the decision boundary of one class without impinging on the decision space of another class. In MNIST, where the decision space for one class is presumably larger, sleep can alter decision boundaries in a way that has a minimal effect on other classes.

Sleep successfully increases the network's robustness to the JSMA attacks on MNIST and Patches, reducing the attack success rate in the case of MNIST and increasing the distance needed to create an adversary for Patches. On CUB-200, there is a marginal reduction in the adversarial attack accuracy compared to the control network. Defensive distillation and fine-tuning also reduce JSMA's effectiveness. However, for these two defenses, in the case of MNIST, the networks were capable of finding an adversary for a higher percentage of the testing set. Thus, the effect of changing a small number of important pixels is mitigated after running the sleep algorithm.

For the Boundary Attack, we found that no defense mechanism helps compared to the control in decreasing the attack's effectiveness on the MNIST dataset. However, for CUB-200 and Patches, the sleep algorithm results in a higher defense score than that for the control network. This lends support to the idea that sleep favorably alters decision boundaries so that it becomes harder to find an adversarial example that is close to the original image after the sleep phase. This also suggests that sleep is not simply obfuscating gradients, which has been a common criticism of several adversarial defenses (Athalye, Carlini, and Wagner 2018), which are tested on white-box attacks. In fact, given the long run-time for convergence of this algorithm, if we define a

threshold for adversarial attack success (L2-norm > 1), then sleep successfully defends against this attack on the MNIST dataset (see Table 3.4).

Why does sleep phase help? It has been shown that sleep tends to promote an increase in stronger weights while pruning weaker weights, thus increasing the width of the weights' distribution (Gonzalez et al. 2019). This results in the consolidation of strong memories at the cost of diminishing weak memories. From this point of view, a memory is a subspace or abstraction in the decision space corresponding to a given class. Sleep may result in enlarging the subspace the network allocates to a stronger category while shrinking weaker ones (Figure 3.6A). The process of strengthening the strong memory also results in making it robust and noise invariant, as seen in Figure 3.6B where the first 8 categories (numbers 0-7) are strengthened and become more invariant to the FGSM attack, while the last two digits are essentially forgotten and the network cannot confidently predict exemplars from these classes (Figure 3.6C). If the noise is less targeted, as in the case of random noise or blurring, sleep does not need to alter the decision space as much to produce better generalization and can maintain a high baseline accuracy, as we demonstrate in the next section.

3.6.2 Distortions

Figure 3.3 shows the network performance for noisy and blurry distortions of data for MNIST (A) as well as noisy distortions for the CUB-200 feature embeddings (B, see Figure 3.4 for results on Patches). Overall, fine-tuning on an image distortion results in the best performance for that specific distortion. However, as was noted (Geirhos et al. 2018), fine-tuning on a specific distortion does not extend to other types of distortions. In our analysis, fine-tuning the network on blurred MNIST images results in high performance (>80%) on blurred images. However, for noisy images, this performance was only marginally above the control network. The sleep algorithm

increased performance for both distortion methods, since this approach is not tailored to any one representation of the training set.

Finally we tested how sleep increases robustness on blur and noise distortions. In biological systems, sleep increases generalization through replay of memories learned during awake which leads to changes in synaptic weights. These changes entail both an increase in synaptic weights associated with a specific task and pruning of synapses involved in other tasks (Gonzalez et al. 2019; Tononi and Cirelli 2006). Figures 3.12 and 3.13 show that correlations among like digits in the hidden layers of the network are greater after applying sleep than before for noisy and blurred images. Likewise, pairs of different digits usually become decorrelated after sleep, suggesting synaptic pruning. We also show that both normalized spiking activity and activations of digit-specific neurons are higher after sleep than before (Figures 3.12 and 3.13, see Appendix for details). These results suggest that the sleep algorithm increases robustness through biologically plausible learning mechanisms involving replay of relevant activity during sleep phase.

3.7 Conclusions and Future Directions

In this work, we show that a biologically inspired sleep algorithm can increase an ANN's robustness to both adversarial attacks and general image distortions. The algorithm augments the normal (e.g., back-propagation based) training phase of an ANN with an unsupervised learning phase in the equivalent SNN modelled after how the biological brain utilises sleep to improve learning. We hypothesize that the unsupervised sleep phase creates more natural feature representations which in turn lead to more natural decision boundaries, thus increasing the robustness of the network. Although this robustness may come at a cost of overall accuracy, it has been shown that robustness may have multiple important benefits, such as more salient feature representations as well as invariance to input modifications (Tsipras et al. 2018). We also show

that the trade-off between robustness and accuracy does not always occur, particularly for image distortions such as noise or blur. Future work includes converting the sleep algorithm into a regularization technique to be applied in more standardized machine learning frameworks as well as understanding the theoretical basis for the beneficial role of spike based plasticity rules in increasing network robustness.

3.8 Acknowledgements

This work was supported by the Lifelong Learning Machines program from DARPA/MTO (HR0011-18-2-0021) and ONR (MURI: N00014-16-1-2829). Chapter 3, in full, is a reprint of the material as it was published in the International Conference of Learning Representations 2020. Tadros, Timothy; Krishnan, Giri; Ramyaa, Ramyaa; Bazhenov, Maxim. ICLR, 2020 The dissertation author was one of the primary investigators and first author on this paper.

3.9 Appendix

3.9.1 Training Parameters

Here, we define the neural network parameters used for each of the three datasets as well as the sleep, defensive distillation, and fine-tuning parameters. Table 3.2 shows parameters used to train each of the control networks discussed in the paper. All neural networks were trained with ReLU neurons. Table 3.3 shows the parameters used during sleep for each of the three datasets. Note that these parameters for MNIST and CUB-200 were chosen by running a genetic algorithm to maximimize performance on the FGSM attack (performance was determined based on the training set so as not to overfit to the test set). For the other three attacks, parameters that maximized FGSM performance were used. Also, for noise and blur generalization, different parameters were chosen (not shown here).

Table 3.2: Parameters used to train the control network for each of the three datasets.

Architecture refers to number of units per layer. For example, the MNIST network possessed 1 input layer, 2 hidden layers with 1200 units, and an output unit with 10 units.

	Patches	MNIST	CUB200
Architecture	[100, 4]	[784, 1200, 1200, 10]	[2048, 350, 300, 200]
Learning Rate	0.1	0.1	0.1
Momentum	0.5	0.5	0.5
Dropout	0	0.2	0.25
Epochs	1	2	100

Table 3.3: Parameters used during sleep

Input rate = F_{max}, the maximum firing rate of input neurons, Sleep duration = length of sleep (number of images presented during sleep, Thresholds = neuronal firing thresholds for each layer of neurons, Synaptic AMPA current = factor to scale the weights by during sleep, Increase and Decrease factor = amount weights are modified on a STDP event.

	Patches	MNIST	CUB200
Input Rate	16 Hz	40 Hz	79 Hz
Sleep Duration	3000	27105	11751
Thresholds	1.0450, 0.7150, 0.3850	36.18, 23.36, 36.38	2.69, 4.61, 2.63
Synaptic AMPA current	4.25	2.19	4.15
Increase factor	0.0035	0.063	0.0016
Decrease factor	0.0002	0.069	0.000209

For the defensively distilled networks tested in the paper, we first train an initial network using a temperature of 50. Then, we use the training set to compute soft labels and finetune the initial network on these soft labels for the same number of epochs and with the same learning rate.

For the fine-tuned networks, we take the control networks trained with the parameters shown in Table 3.1. The learning rate is reduced to 0.05 and the network is fine-tuned on a mixture of either adversarial attacks, blur or noise and the original images/features. For CUB-200, we perform fine-tuning for 10 epochs.

The Patches dataset represents an easily interpretable example where we can understand what happens to the weights after sleep. Figure 3.4A shows an example of the dataset. Here, we have 4 images each belonging to 4 different classes. 25 pixels are whitened in each image and the remaining 75 pixels are dark. There is a 15 pixel overlap, so that weights connecting from input to output layer must take this into account in order to separate the images. Figure 3.4B illustrate the blur and noise distortions tested for this dataset and Figure 3.4C shows the results for the blur and noise distortions.

After the network is trained, we can analyze the weights connecting from each of the 100 input neurons to the 4 output neurons (see Figure 3.5, top row). We theorize that optimally robust behavior would occur when weights connecting from ON-pixels are positive, weights connecting from overlapping pixels are near 0, and weights connecting from OFF-pixels are negative. In this case, changing the value of overlapping pixels will have no effect on classification. Changing the value of OFF-pixels will cause the network to predict another class, where OFF-pixels may be ON-pixels or indicative of that class. Changing the value of ON-pixels will only have a negative impact if the brightness of the pixel is reduced significantly. Thus, in this circumstance, the network should behave robustly.

In the control network, we observe that weights connecting from ON-pixels (pixel-value = 1) increase while weights connecting from OFF-pixels remain at 0. Weights connecting from overlapping pixels remain near 0 or positive. Defensive distillation causes some weights connecting from overlapping pixels to decrease, likely because the soft labels used in defensive distillation cause overlapping pixel units to alter the probability values computed by the network in such a way that does not truly reflect the impact of the overlapping pixels. In the fine-tuned networks (both on blurred images and noisy images), we observe an increase in ON-pixel weights

and an increase in noisiness of OFF-pixel weights. Likewise, in the sleep network, OFF-pixel weights become negative while ON-pixel weights remain the same. In these cases, robustness is increased as weights become more similar to our hypothesized ideal weights. Essentially, the magnitude of input changes need to change classification increase since the spread between ON-pixel weights and OFF-pixel weights increases. We quantify the spread in weights by taking the difference between the average weight connecting from ON-pixels and the average weight connecting from OFF-pixels. This represents the mean input that each correct output neuron receives. This result is shown in Figure 3.4D. Of note is that this weight spread is increased for both the sleep and finetuning-noise network, suggesting that these defenses bring the weights closer to their ideal values for computing robustness.

3.9.1 Adversarial Attacks

Here, we describe the general approach for implementing DeepFool, JSMA, and the Boundary Attack discussed in the paper. We also show examples of adversaries created for each of the defense networks from these attacks.

3.9.1.1 DeepFool

DeepFool (Moosavi-Dezfooli, Fawzi, and Frossard 2016), as mentioned above, is an iterative algorithm that, at each iteration, aims to move the adversarial example in the direction of the closest decision boundary until it results in a misclassification. We based our implementation of that in (Rauber, Brendel, and Bethge 2017). We stopped running the algorithm when either an adversarial example is found or when 100 iterations have passed. Examples of DeepFool attacks on the MNIST dataset are shown in Figure 3.7A. At each iteration we compute a linear approximation of the loss function and take a step in the direction that would be result in a misclassification. The equations used and pseudocode can be found in the original DeepFool paper.

3.9.1.2 JSMA

JSMA is also an iterative algorithm which computes the pixel that would change the loss function the most at each algorithm and changes this pixel, until a misclassification is produced. For this method, we set a run-time limit of 500 iterations. We also remove a pixel from the saliency map when it has been updated seven times, so the algorithm can focus on other pixels. We set the change to each pixel at a constant value, 0.1. This represents how much each pixel is updated (in the direction that results in a misclassification) at each iteration. Pseudocode can be found in the original publication (Papernot, McDaniel, Jha, et al. 2016). We show examples of adversaries created by JSMA in Figure 3.7B.

3.9.1.3 Boundary Attack

The Boundary Attack (Brendel, Rauber, and Bethge 2017) starts with an adversarial example and moves it closer to the decision boundary of the correct class. At each step of the algorithm, the method performs orthogonal and forward perturbations to move the adversary closer to the original image, thus reducing the distance between the adversary and the original image. We set both a distance convergence criterion (L2-norm = 1e-7) and a run-time limitation on the attack (1000 iterations). Example attacks are shown in Figure 3.7C. We note that sometimes the algorithm does not successfully produce an "imperceptible" adversarial example and instead produces a noisy output (the starting condition is a noisy image). If we define a threshold defining a successful adversarial attack (L2-Norm > 1), then we observe the results for MNIST in Table 3.4.

Table 3.4: MNIST Boundary Attack with a threshold defining a successful adversarial attack

MNIST	Control	Defensive Distillation	Fine-tuning	Sleep
Boundary Attack	0.0073	0.0074	0.0047	**0.0094**

3.9.2 Generalization Analysis

In this section, we analyze how sleep can aid in increasing ANN robustness. In biological networks, sleep extracts the gist of a task through replay (Lewis and Durrant 2011). We hypothesized that our sleep algorithm works in the same manner. First, we tested the ability of the sleep network to decorrelate distinct inputs by analyzing the effect of running sleep and testing on our two distortion techniques (see Figure 3.8).

We computed the correlations of network activities in each of the hidden layers of the network before and after implementing our defense methods. For each pair of digits, we computed the average correlation of layer activities in the undistorted (Figure 3.9) , noisy (Figure 3.10) and blurred (Figure 3.11) conditions. Each figure reports the difference in digit pairwise correlations between the defense method and the control network for each set of inputs. For our sleep network, it is apparent that in layer 2 and layer 3, the correlations of the same digits (the diagonal) increases after sleep. Additionally, the correlation of distinct digits typically experiences negative change, representing decorrelation of distinct inputs. This analysis holds for defensive distillation and both of the fine-tuned networks. This suggests that the ANN representation of different exemplars of the same digit becomes more similar after sleep or after any of the defense networks when compared to the control. This is not simply due to an increased overlap of all inputs, since exemplars of different digits become decorrelated after applying a defense method.

Next, we performed the same correlation analysis on noisy and blurred images to see how the representation of distorted images changes after applying a distortion method. First, we note that fine-tuning on noisy images results in stronger correlation of the same (noisy) digit but weaker correlations of different (noisy) digits, as noted above. However, fine-tuning on blurred images does not have as strong an effect. Second, sleep seems to have a beneficial effect on the correlation matrices for both blurred and noisy images (comparing the right column of Figures 3.12 and 3.13). This illustrates the beneficial role of sleep in creating distinct representations of digits, where different neuronal ensembles encode different digits. This change in representation should result in increased robustness since changes to the input must be larger in order to recruit neuronal ensembles that represent other digits.

On top of decorrelating the representation of distinct memories by pruning synapses, biophysical modelling suggests that sleep can also aid in strengthening connections thus making stronger the response of primary neurons involved in memory recall (Gonzalez et al. 2019). To test this hypothesis in our networks, we analyzed the firing rate and activations of digit-specific neurons before and after sleep. Before describing the analysis, we would like to note that SNNs can be used to performn classification and a near loss-less conversion between ANNs and SNNs has been achieved on the MNIST task (P. U. Diehl et al. 2015). To perform classification, a digit is presented (as a Poisson spike train) to the network and spikes are propagated throughout the network for a given time period (or number of presentations of the input). Analyzing network activity in the spiking domain can be easier than in the activation domain (ANNs) since spikes are oftentimes easier to interpret than neuronal activations.

For this reason, we first analyze how spike rates of digit-specific neurons change before and after sleep in the spike domain. To do this we present all images of a specified digit to the

spiking network and count the number of spikes from each neuron (holding the weights constant). We define digit-specificity by looking at the 100 neurons with the highest firing rates in layer 2. In Figure 3.12, we show that the normalized firing rate of these neurons usually increases after sleep (normalized by dividing by the maximum firing rate observed from the SNN).

Next, we perform the same analysis in the activation domain. Again, we define digit-specific neurons by looking at the top 100 neurons with the highest activation for a specific digit. We look at the normalized mean activations of these neurons before and after sleep and note that for all digits this value is higher after sleep than before sleep (Figure 3.13). This suggests that the neurons in the network are responding more strongly to the presentation of the same digit, thus increasing the robustness of the network as more noise must be added in order to counter the effect of this stronger response. This also suggests that our algorithm works in a biologically plausible way: both by decorrelating distinct inputs and increasing the strength of similar inputs.

3.10 Figures

Algorithm 1 Sleep:

procedure CONVERTANNTOSNN(nn)
    Map the weights from (nn) with ReLU units to network of integrate-fire units (snn)
    Apply weight normalization and return scale for each layer ([24]) **return** $snn, scales$
procedure CONVERTSNNTOANN(nn)
    Directly map the weights from integrate-fire network (nn) to ReLU network (ann) **return** ann
procedure SLEEP($nn, I, scales$) ▷ I is input
    Initialize v (voltage) $= 0$ vectors for all neurons
    for $t \leftarrow 1$ to Ts **do** ▷ Ts - duration of sleep
        $S(1,t) \leftarrow$ Convert input I to Poisson-distributed spiking activity
        for $l \leftarrow 2$ to n **do** ▷ n - number of layers
            $v(l,t) \leftarrow v(l,t-1) + (scales(l-1)\mathbf{W}(l,l-1)S(l-1,t))$ ▷ W(l,l-1) - weights
            $S(l,t) \leftarrow v(l,t) > threshold(l)$ ▷ Propagate spikes
            $\mathbf{W}(l,l-1) \leftarrow \begin{cases} \mathbf{W}(l,l-1) + inc & \text{if } S(l,t) = 1 \,\&\, S(l-1,t) = 1 \\ \mathbf{W}(l,l-1) - dec & \text{if } S(l,t) = 1 \,\&\, S(l-1,t) = 0 \end{cases}$ ▷ STDP
procedure MAIN
    Initialize neural network (ann) with ReLU neurons and bias $= 0$.
    Train ann using backpropagation.
    $snn, scales$ = ConvertANNtoSNN(ann)
    snn = Sleep(snn, Training data X, $scales$)
    ann = ConvertSNNtoANN(snn)

Figure 3.1: Sleep Replay Algorithm for increased generalization and robustness

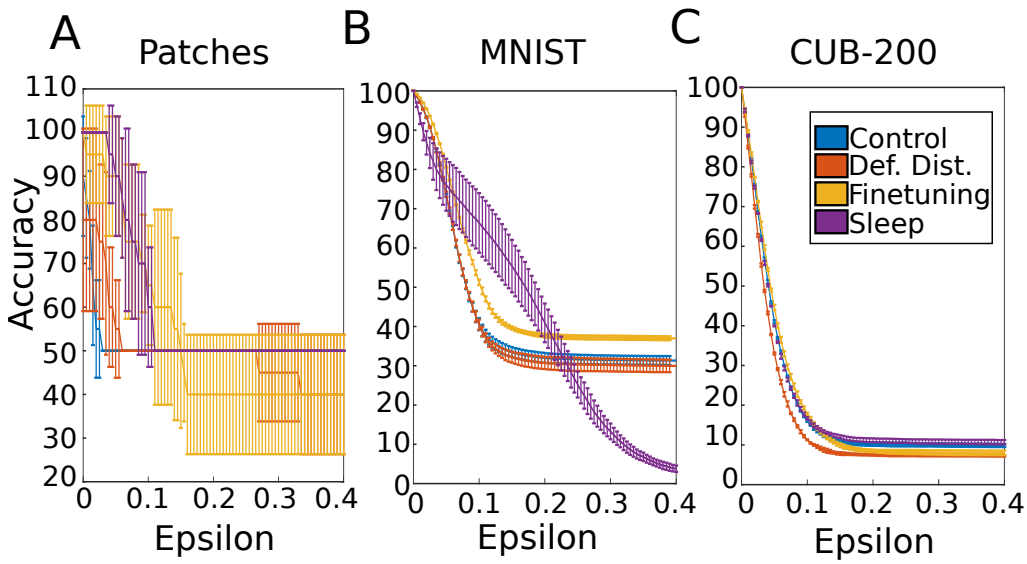

Figure 3.2: FGSM classification accuracy as a function of noise added for three datasets.

(A) Patches
(B) MNIST
(C) CUB-200

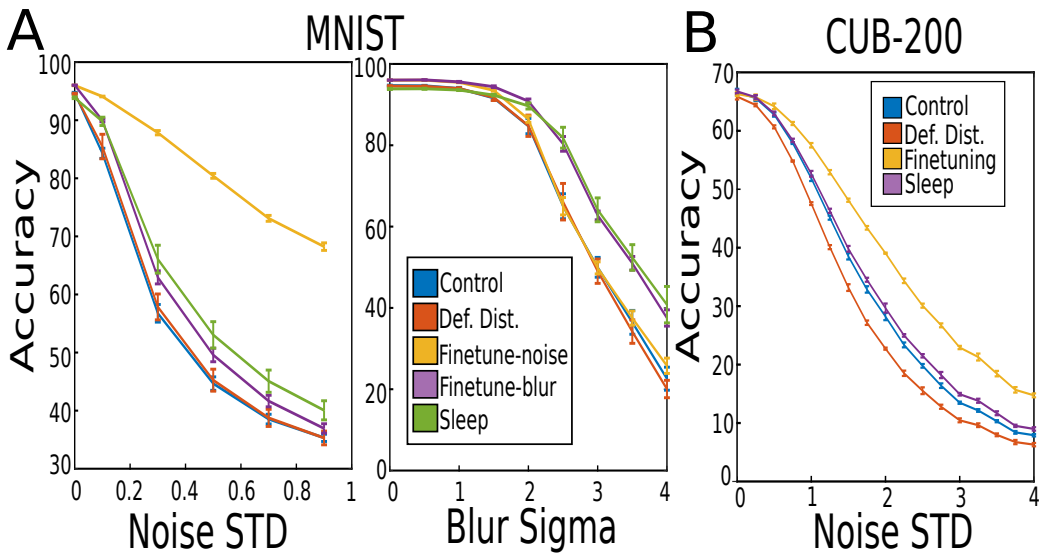

Figure 3.3: Sleep increases robustness to general distortions

(A) Generalization classification accuracy for 5 networks for noise and blur on the MNIST dataset.
(B) Generalization classification accuracy for 4 networks for the noisy CUB-200 task. Note that there are only 4 networks because there is no blur task here.

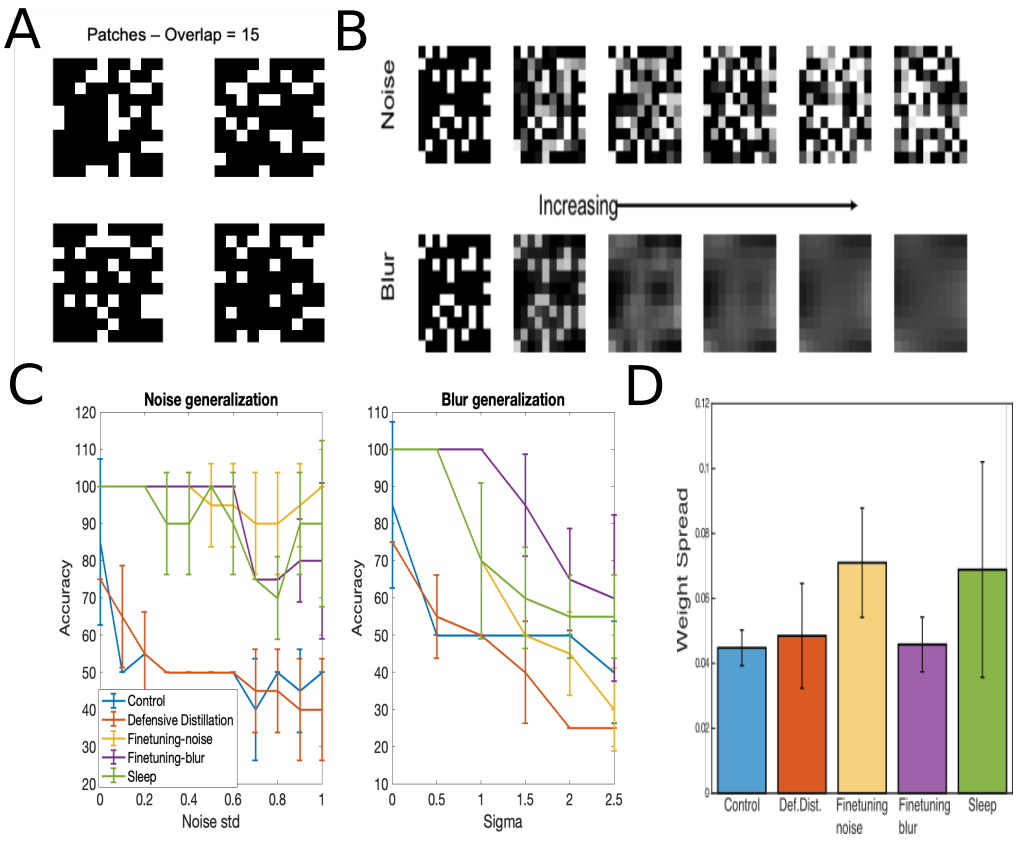

Figure 3.4: Patches dataset analysis

(A) Patches dataset example – four binary images with 25 pixels turned on in each image and 15 pixel overlap.
(B) Types of images tested on for generalization for the Patches dataset. Top – images with Gaussian noise added with increasing variance (from 0 to 1.0 in steps of 0.2). Bottom – Gaussian blurred images with increasing sigma (from 0 to 2.5 in steps of 0.5).
(C) Generalization accuracy for noise and blur of five different networks tested (Control, Defensively distilled, Fine-tuned-blur, fine-tuned-noise, and Sleep) for the Patches Dataset.
(D) Weight spread for each of the 5 networks tested in C.

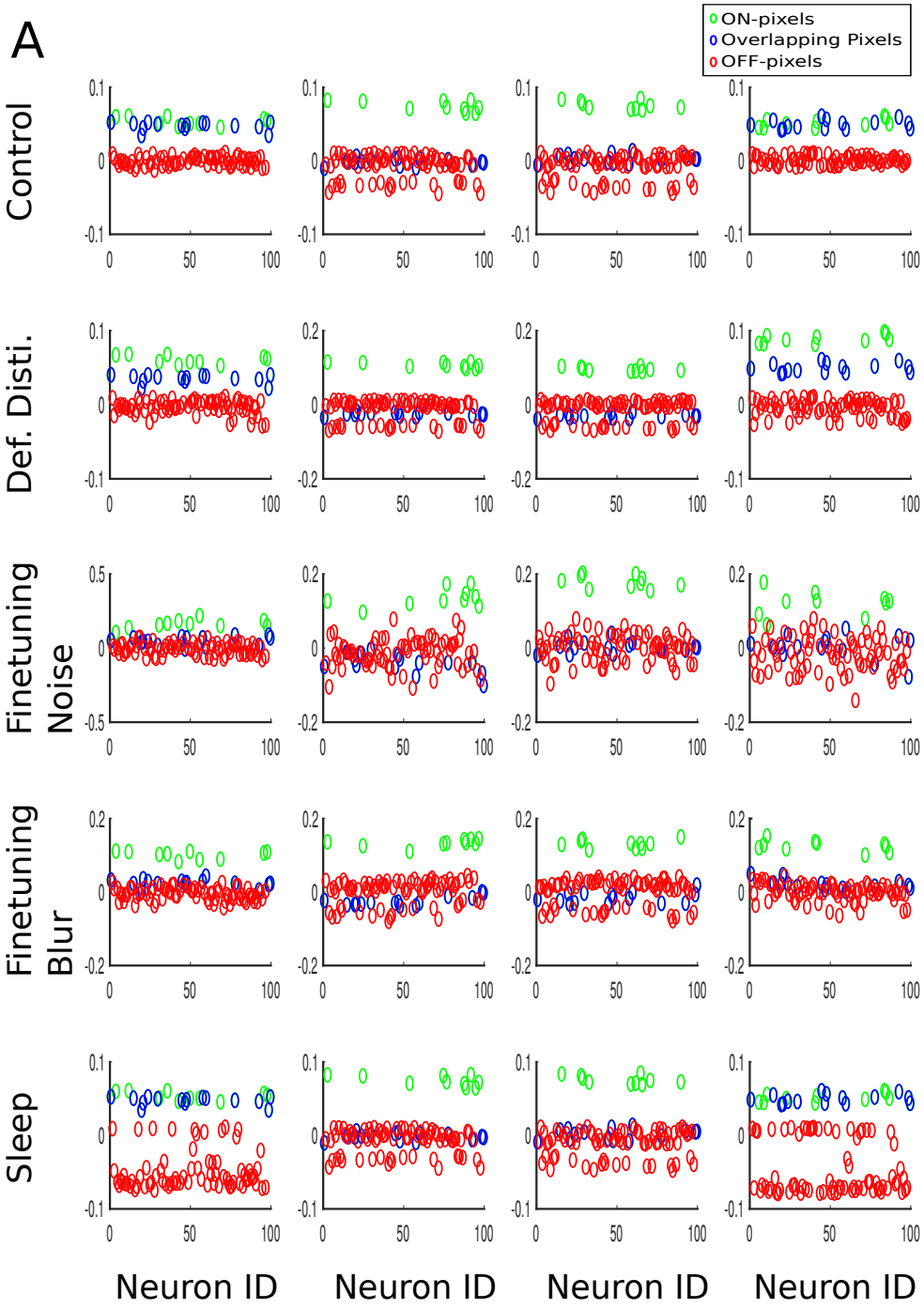

Figure 3.5: Patches weight analysis

Weights for each of the 5 networks (Control, defensively distilled, finetuning on noise, finetuning on blur, and sleep) trained on the Patches dataset. Each column shows the weights value (y-axis) connecting from each of the 100 input neurons (x-axis) to the corresponding output neuron (i.e. the first column of graphs if the weights connecting from all input neurons to the first output neuron. Points are color-coded based on which of these pixels in the input layer correspond to ON-pixels, OFF-pixels, or overlapping pixels based on the input that should trigger that output neuron.

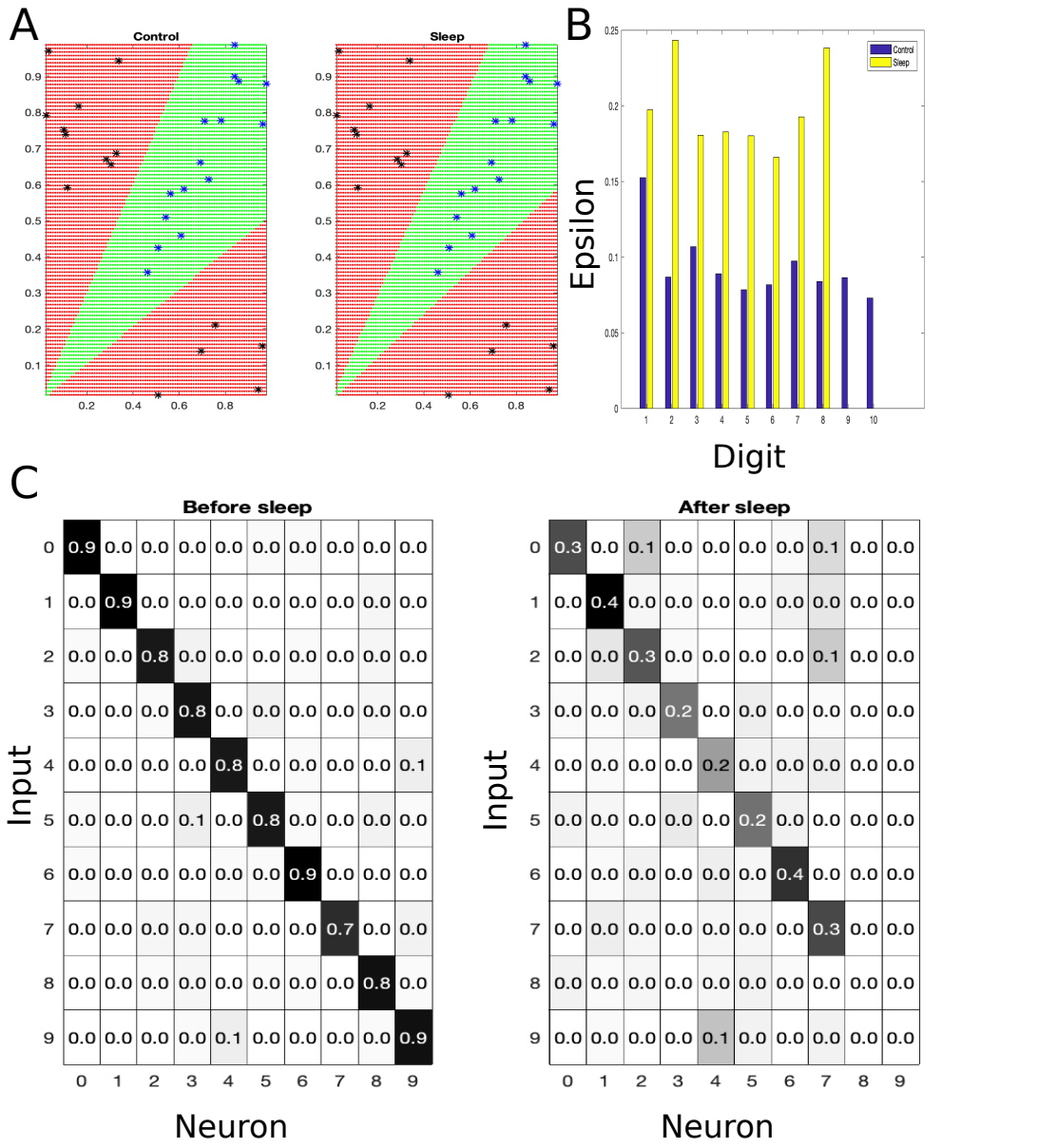

Figure 3.6: Binary classification example

(A) Example function learned in a 3-layer neural network illustrates that sleep alters decision boundaries in favor of making one class (corresponding to black points) more robust while impinging on another class (blue points).
(B) Average noise (epsilon) needed for FGSM attack for specific digits in MNIST dataset.
(C) Output layer scores for each digit (rows) before and after sleep. Column represents average activation of each of the 10 output neurons.

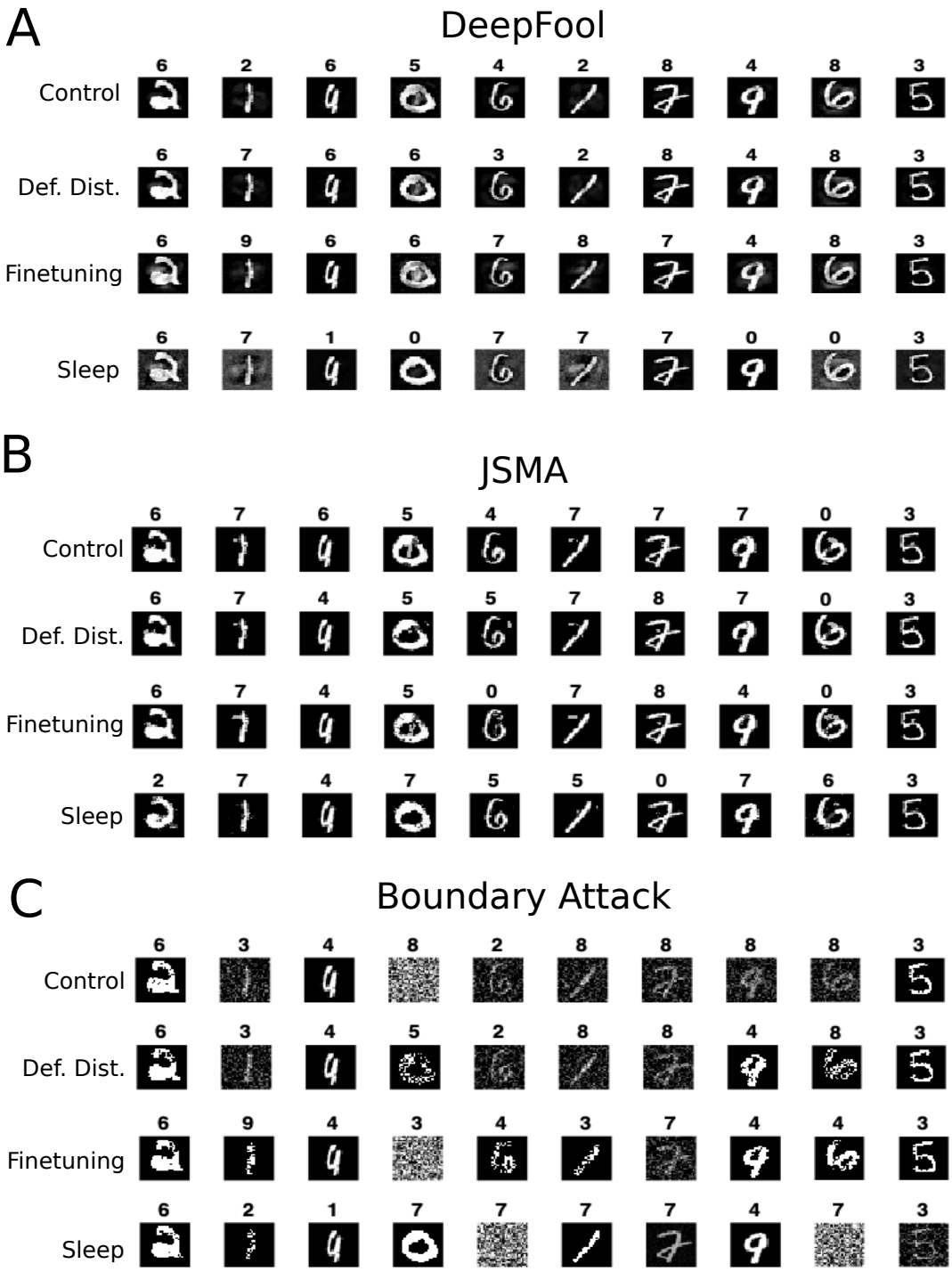

Figure 3.7: Example adversarial attacks

(A) DeepFool adversarial examples for each defense. The network's prediction is shown above each image.
(B) JSMA
(C) Boundary Attack

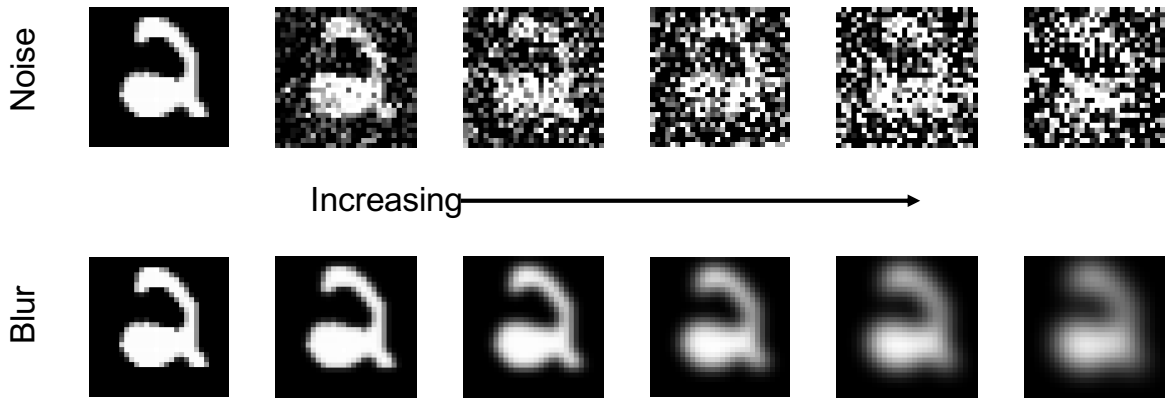

Figure 3.8: Types of images tested on for generalization for the MNIST dataset.

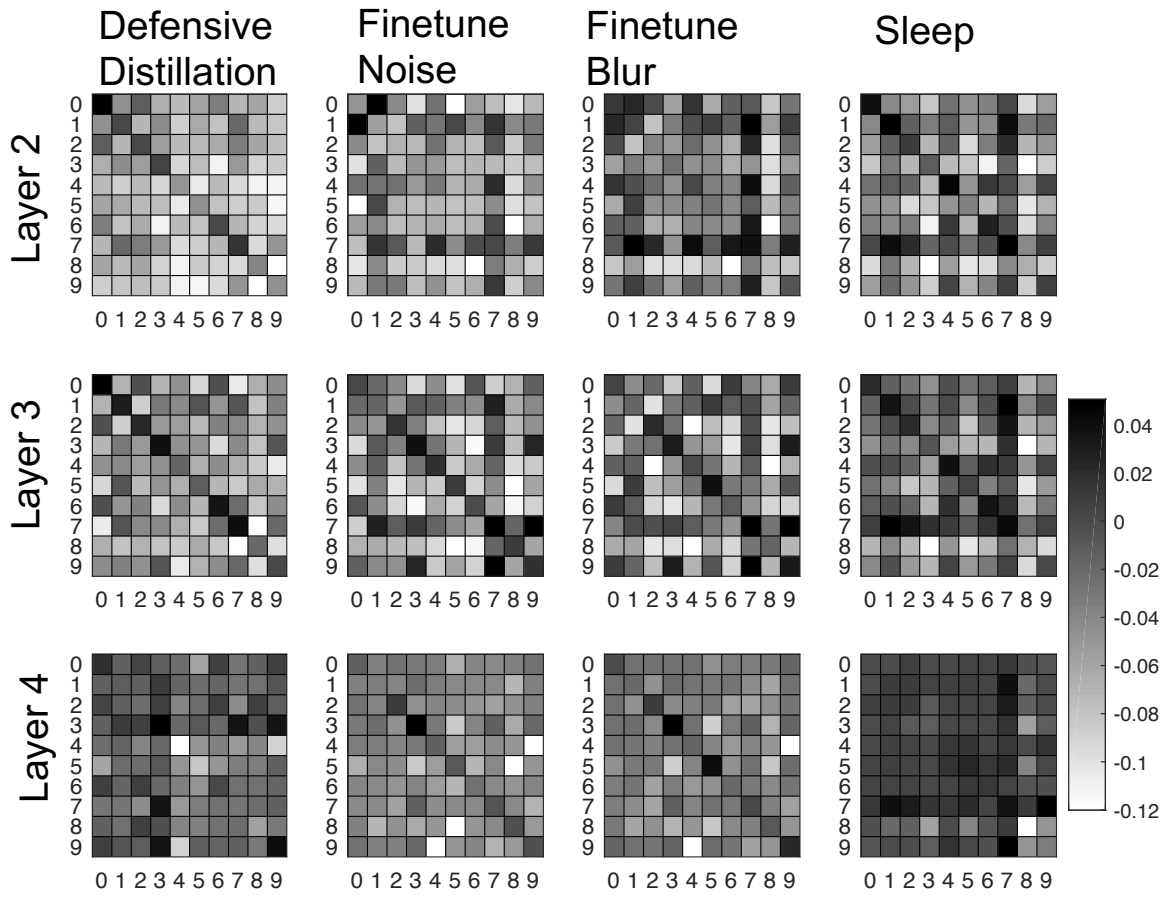

Figure 3.9: correlation differences between defense network and control network for 4 different defenses

Correlations are computed based on the activations in each layer for each pair of digits (mean correlation). The difference between the correlation of the defense method (column) and the control network is plotted. Activations are computed based on undistorted test images.

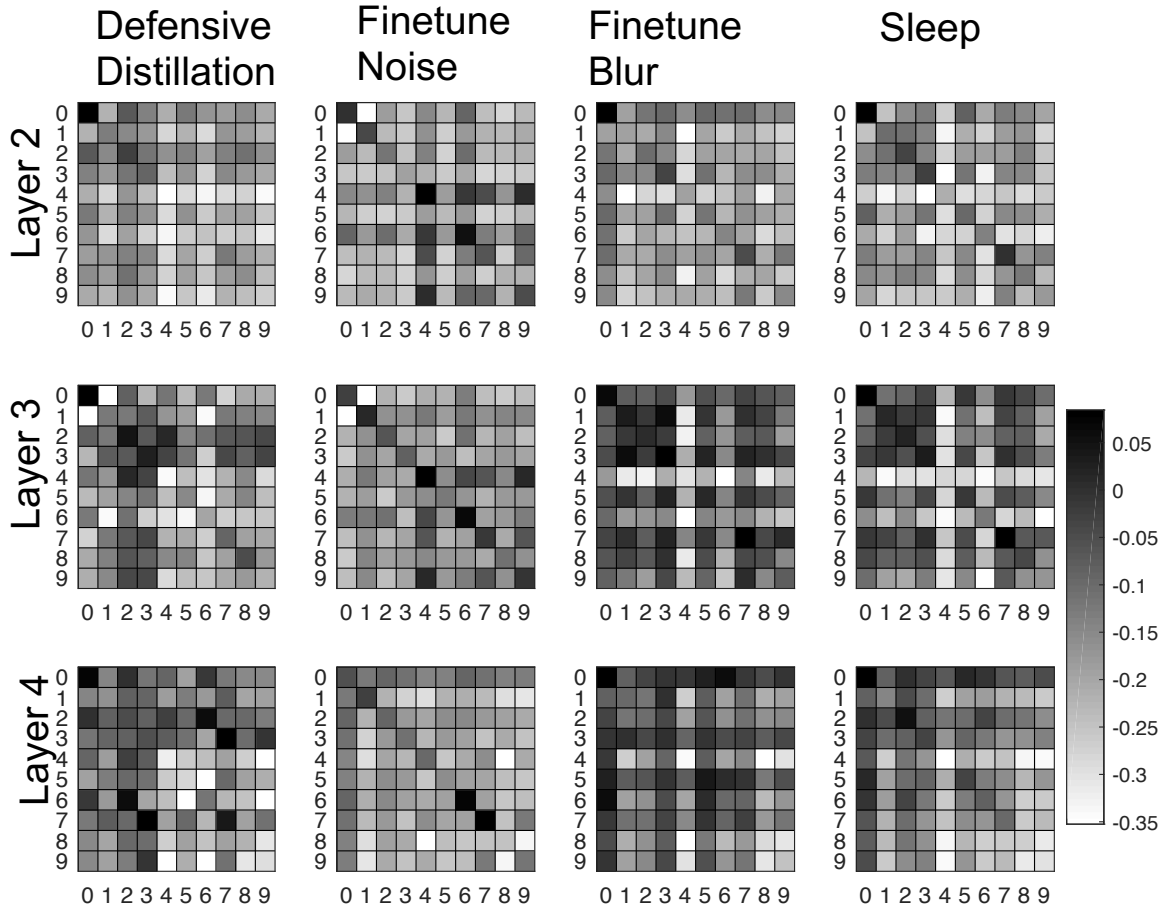

Figure 3.10: Correlation differences on noisy images

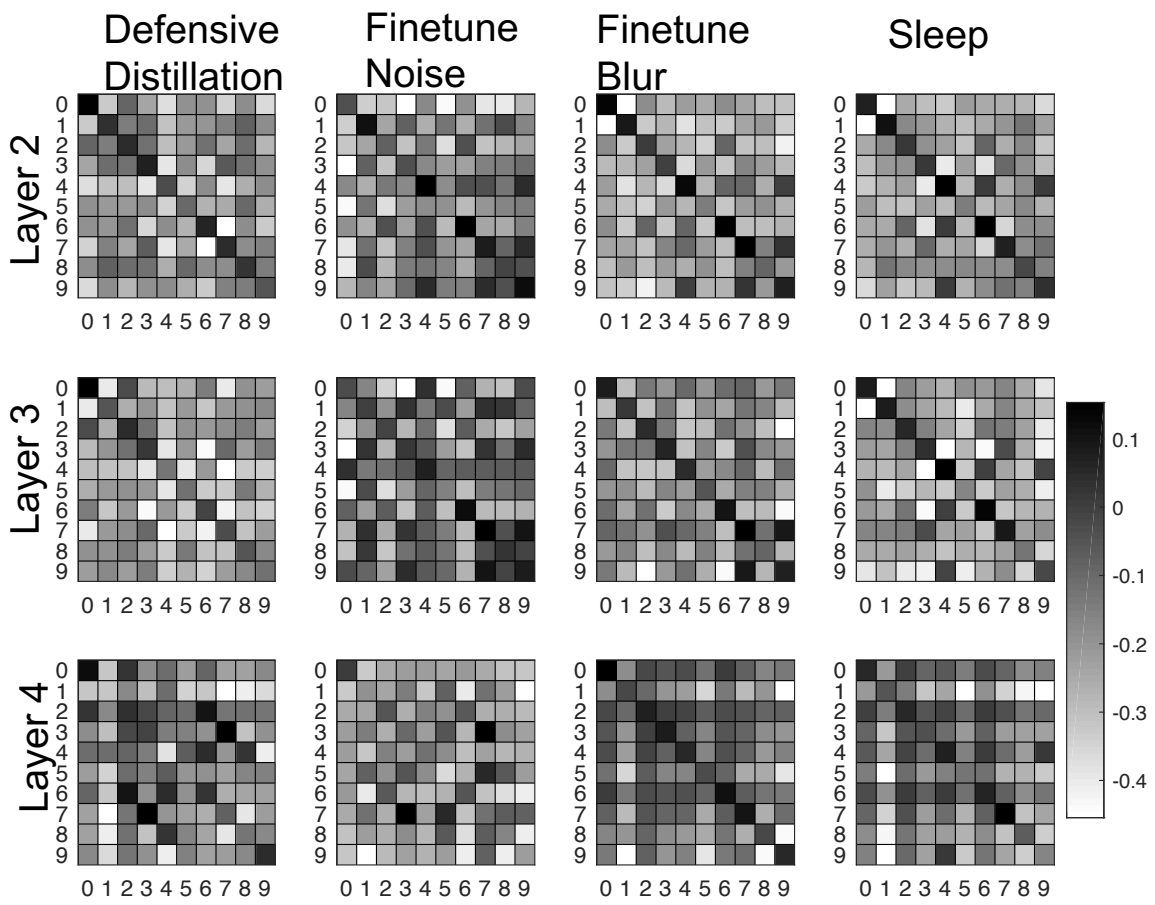

Figure 3.11: Correlation differences on blurred images

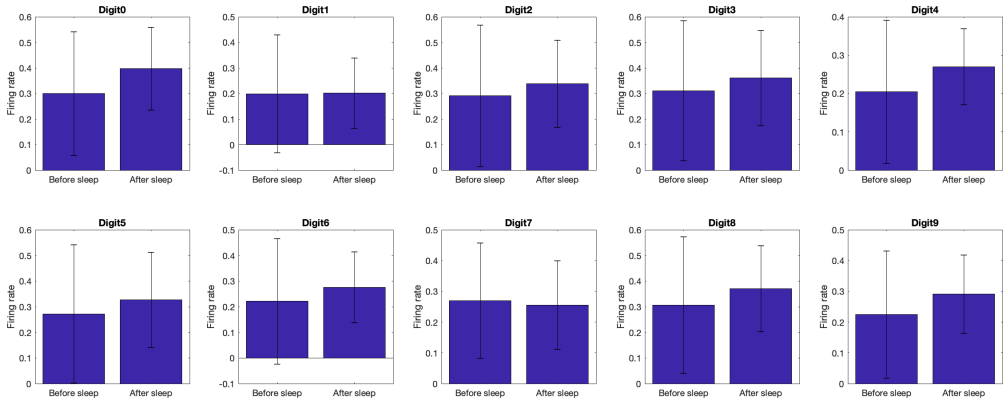

Figure 3.12: Normalized firing rates of neurons specific to individual digits when presented with noisy images is greater after applying sleep than before sleep.

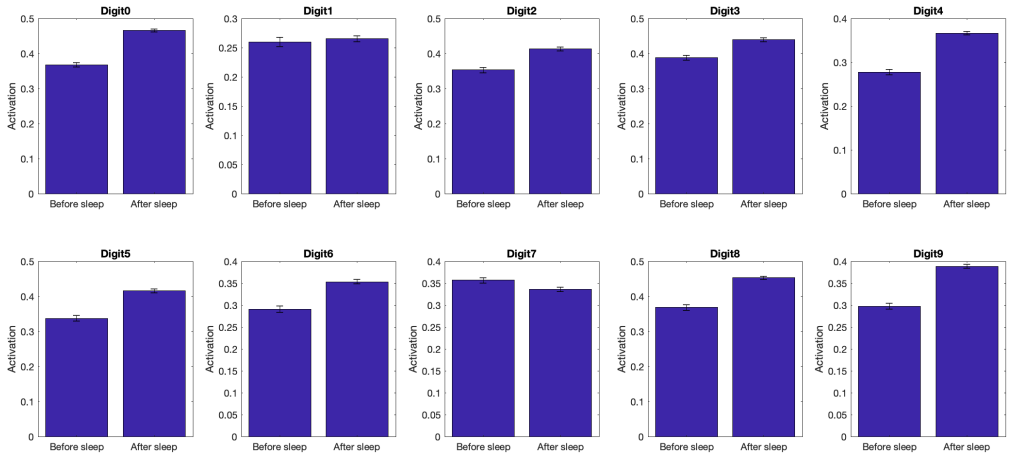

Figure 3.13: Normalized activations of neurons specific to individual digits when presented with noisy images is greater after applying sleep than before sleep.

3.11 References

Abbott, LF, and Thomas B Kepler. 1990. "Model Neurons: From Hodgkin-Huxley to Hopfield." In *Statistical Mechanics of Neural Networks*, 5–18. Springer.

Athalye, Anish, Nicholas Carlini, and David Wagner. 2018. "Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples." *ArXiv Preprint ArXiv:1802.00420*.

Athalye, Anish, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. 2017. "Synthesizing Robust Adversarial Examples." *ArXiv Preprint ArXiv:1707.07397*.

Biggio, Battista, and Fabio Roli. 2018. "Wild Patterns: Ten Years after the Rise of Adversarial Machine Learning." *Pattern Recognition* 84: 317–31.

Brendel, Wieland, Jonas Rauber, and Matthias Bethge. 2017. "Decision-Based Adversarial Attacks: Reliable Attacks against Black-Box Machine Learning Models." *ArXiv Preprint ArXiv:1712.04248*.

Carlini, Nicholas, and David Wagner. 2016. "Defensive Distillation Is Not Robust to Adversarial Examples." *ArXiv Preprint ArXiv:1607.04311*.

———. 2017. "Towards Evaluating the Robustness of Neural Networks." In *2017 IEEE Symposium on Security and Privacy (SP)*, 39–57. IEEE.

Diehl, Peter U, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. 2015. "Fast-Classifying, High-Accuracy Spiking Deep Networks through Weight and Threshold Balancing." In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Dodge, Samuel, and Lina Karam. 2016. “Understanding How Image Quality Affects Deep Neural Networks.” In *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, 1–6. IEEE.

———. 2019. “Human and DNN Classification Performance on Images With Quality Distortions: A Comparative Study.” *ACM Transactions on Applied Perception (TAP)* 16 (2): 7.

Ford, Nic, Justin Gilmer, Nicolas Carlini, and Dogus Cubuk. 2019. “Adversarial Examples Are a Natural Consequence of Test Error in Noise.” *ArXiv Preprint ArXiv:1901.10513*.

Geirhos, Robert, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. 2018. “Generalisation in Humans and Deep Neural Networks.” In *Advances in Neural Information Processing Systems*, 7538–50.

Ghosh-Dastidar, Samanwoy, and Hojjat Adeli. 2009. “Spiking Neural Networks.” *International Journal of Neural Systems* 19 (04): 295–308.

Gonzalez, Oscar C, Yury Sokolov, Giri Krishnan, and Maxim Bazhenov. 2019. “Can Sleep Protect Memories from Catastrophic Forgetting?” *BioRxiv*, 569038.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy. 2014. “Explaining and Harnessing Adversarial Examples.” *ArXiv Preprint ArXiv:1412.6572*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep Residual Learning for Image Recognition.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–78.

Huang, Ruitong, Bing Xu, Dale Schuurmans, and Csaba Szepesvari. 2015. “Learning with a Strong Adversary.” *ArXiv Preprint ArXiv:1511.03034*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. “Imagenet Classification with Deep Convolutional Neural Networks.” In *Advances in Neural Information Processing Systems*, 1097–1105.

LeCun, Yann, Léon Bottou, Yoshua Bengio, Patrick Haffner, and others. 1998. “Gradient-Based Learning Applied to Document Recognition.” *Proceedings of the IEEE* 86 (11): 2278–2324.

Lewis, Penelope A, and Simon J Durrant. 2011. “Overlapping Memory Replay during Sleep Builds Cognitive Schemata.” *Trends in Cognitive Sciences* 15 (8): 343–51.

Moosavi-Dezfooli, Seyed-Mohsen, Alhussein Fawzi, and Pascal Frossard. 2016. “Deepfool: A Simple and Accurate Method to Fool Deep Neural Networks.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2574–82.

Nguyen, Anh, Jason Yosinski, and Jeff Clune. 2015. “Deep Neural Networks Are Easily Fooled: High Confidence Predictions for Unrecognizable Images.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 427–36.

Papernot, Nicolas, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and

Ananthram Swami. 2016. “The Limitations of Deep Learning in Adversarial Settings.” In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*, 372–87. IEEE.

Papernot, Nicolas, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. “Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks.” In *2016 IEEE Symposium on Security and Privacy (SP)*, 582–97. IEEE.

Payne, Jessica D, Daniel L Schacter, Ruth E Propper, Li-Wen Huang, Erin J Wamsley, Matthew A Tucker, Matthew P Walker, and Robert Stickgold. 2009. “The Role of Sleep in False Memory Formation.” *Neurobiology of Learning and Memory* 92 (3): 327–34.

Rasch, Björn, and Jan Born. 2013. “About Sleep’s Role in Memory.” *Physiological Reviews* 93 (2): 681–766.

Rauber, Jonas, Wieland Brendel, and Matthias Bethge. 2017. “Foolbox: A Python Toolbox to Benchmark the Robustness of Machine Learning Models.” *ArXiv Preprint ArXiv:1707.04131*.

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den

Driessche, Julian Schrittwieser, et al. 2016. “Mastering the Game of Go with Deep Neural Networks and Tree Search.” *Nature* 529 (7587): 484.

Song, Sen, Kenneth D Miller, and Larry F Abbott. 2000. “Competitive Hebbian Learning through Spike-Timing-Dependent Synaptic Plasticity.” *Nature Neuroscience* 3 (9): 919.

Stickgold, Robert. 2005. “Sleep-Dependent Memory Consolidation.” *Nature* 437 (7063): 1272.

Stickgold, Robert, and Matthew P Walker. 2013. “Sleep-Dependent Memory Triage: Evolving Generalization through Selective Processing.” *Nature Neuroscience* 16 (2): 139.

Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2013. “Intriguing Properties of Neural Networks.” *ArXiv Preprint ArXiv:1312.6199*.

Tononi, Giulio, and Chiara Cirelli. 2006. “Sleep Function and Synaptic Homeostasis.” *Sleep Medicine Reviews* 10 (1): 49–62.

Tsipras, Dimitris, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander

Madry. 2018. “Robustness May Be at Odds with Accuracy.” *ArXiv Preprint ArXiv:1805.12152*.

Uesato, Jonathan, Brendan O'Donoghue, Aaron van den Oord, and Pushmeet Kohli. 2018. "Adversarial Risk and the Dangers of Evaluating against Weak Attacks." *ArXiv Preprint ArXiv:1802.05666*.

Vasiljevic, Igor, Ayan Chakrabarti, and Gregory Shakhnarovich. 2016. "Examining the Impact of Blur on Recognition by Convolutional Networks." *ArXiv Preprint ArXiv:1611.05760*.

Wamsley, Erin J, Matthew A Tucker, Jessica D Payne, and Robert Stickgold. 2010. "A Brief Nap Is Beneficial for Human Route-Learning: The Role of Navigation Experience and EEG Spectral Power." *Learning & Memory* 17 (7): 332–36.

Wei, Yina, Giri P Krishnan, Maxim Komarov, and Maxim Bazhenov. 2018. "Differential Roles of Sleep Spindles and Sleep Slow Oscillations in Memory Consolidation." *PLoS Computational Biology* 14 (7): e1006322.

Welinder, Peter, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. 2010. "Caltech-UCSD Birds 200."

Wilson, Matthew A, and Bruce L McNaughton. 1994. "Reactivation of Hippocampal Ensemble Memories during Sleep." *Science* 265 (5172): 676–79.

Yuan, Xiaoyong, Pan He, Qile Zhu, and Xiaolin Li. 2019. "Adversarial Examples: Attacks and Defenses for Deep Learning." *IEEE Transactions on Neural Networks and Learning Systems*.

Zhou, Yiren, Sibo Song, and Ngai-Man Cheung. 2017. "On Classification of Distorted Images with Deep Convolutional Neural Networks." In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1213–17. IEEE.

4 Sleep-like Unsupervised Replay Reduces Catastrophic Forgetting in Artificial Neural Networks

4.1 Abstract

Artificial neural networks (ANNs) are known to suffer from catastrophic forgetting: when learning multiple tasks sequentially, they perform well on the most recent task at the expense of previously learned tasks. In the brain, sleep is known to play an important role in incremental learning by replaying recent and old conflicting memory traces. Here we tested if implementing sleep-like phase in ANNs can protect old memories during new training and alleviate catastrophic forgetting. Sleep was implemented as off-line training with local unsupervised Hebbian plasticity rules and noisy input. In an incremental learning framework, sleep was able to recover old tasks that were otherwise forgotten. Previously learned memories were replayed spontaneously, forming unique representations for each class of inputs. Representational sparseness and neuronal activity corresponding to the old tasks increased while new task related activity decreased. The study suggests that spontaneous replay simulating brain dynamics can alleviate catastrophic forgetting in ANNs.

4.2 Introduction

Humans and animals have a remarkable ability to learn continuously and to incorporate new data into their corpus of existing knowledge. In contrast, artificial neural networks (ANNs) suffer from "catastrophic forgetting" whereby they achieve optimal performance on newer tasks at the expense of performance on previously learned tasks (McCloskey and Cohen 1989; French 1999; McClelland, McNaughton, and O'Reilly 1995). This dichotomy between continual learning in mammals and catastrophic forgetting in machine learning models has given rise to the stability-plasticity dilemma (Mermillod, Bugaiska, and Bonin 2013). On the one hand, a network must be

plastic such that the parameters in the network can change in order to accurately represent and respond to new tasks. On the other hand, a network must be stable such that it maintains knowledge of older tasks. Although deep neural networks (LeCun, Bengio, and Hinton 2015) can achieve supra-human level of performance on tasks ranging from complex games (Silver et al. 2018) to image recognition (Russakovsky et al. 2015), they lie at a sub-optimal point on the stability-plasticity spectrum.

Sleep has been hypothesized to play an important role in memory consolidation and generalization of knowledge in biological systems (Ji and Wilson 2007; Walker and Stickgold 2004; Lewis and Durrant 2011). During sleep, neurons are spontaneously active without external input and generate complex patterns of synchronized activity across brain regions (Steriade, McCormick, and Sejnowski 1993; Krishnan et al. 2016). Two critical components which are believed to underlie memory consolidation during sleep are spontaneous replay of memory traces and local unsupervised synaptic plasticity (Wilson and McNaughton 1994; Stickgold 2005b; Wei, Krishnan, and Bazhenov 2016). Replay of recently learned memories along with relevant old memories (Rasch and Born 2013; Lewis, Knoblich, and Poe 2018b; Hennevin et al. 1995; Mednick et al. 2011; Paller and Voss 2004; Oudiette et al. 2013) enables the network to form orthogonal memory representations to enable coexistence of competing memories within overlapping populations of neurons (Wei, Krishnan, and Bazhenov 2016; Wei et al. 2018; Golden et al. 2020; Gonzalez et al. 2019). Local plasticity allows synaptic changes to affect only relevant memories. While consolidation of declarative memories presumably depends on the interplay between fast-learning hippocampus and slow-learning cortex (Rasch and Born 2013) ('Complementary Learning Systems Theory' (McClelland, McNaughton, and O'Reilly 1995)), several types of procedural memories (e.g., skills) are believed to be hippocampus-independent and still require

consolidation during sleep, particulartly during Rapid Eye Movement (REM) sleep (McDevitt, Duggan, and Mednick 2015). These results from neuroscience suggest that sleep replay principles applied to ANNs may reduce catastrophic forgetting in machine learning models. In this new study, we focus on the hippocampus-independent consolidation of memories during REM sleep-like activity.

We show that implementing a sleep-like phase after an ANN learns a new task enables replay and makes possible continual learning of multiple tasks without forgetting. These results are formalized as a sleep replay algorithm (SRA) as follows. First, an ANN is trained using the backpropagation algorithm, denoted below as awake training. Next, spontaneous brain dynamics, similar to those found in sleep (Krishnan et al. 2016; Bazhenov et al. 2002), are simulated and we run one-time step of network simulation propagating spontaneous activity forward through the network. Next, we do a backward pass through the network in order to apply local Hebbian plasticity rules to modify weights. After running multiple steps of this unsupervised training phase, testing or further training using regular backpropagation is performed. We recently found that this sleep algorithm can promote domain generalization and improve robustness against adversarial attacks (Tadros et al. 2019). Here, we expand the algorithm to continual learning problem by limiting the amount of information available during sleep. We show that spontaneous reactivation of neurons during sleep engages local plasticity rules that can recover performance on tasks that were thought to be lost due to catastrophic forgetting after new task training.

4.3 Methods

4.3.1 Task Protocols

To demonstrate catastrophic forgetting, we utilized an incremental learning framework, where groups of classes are learned in a sequential fashion. We utilized 5 datasets to illustrate the

prevalence of catastrophic forgetting as well as the beneficial role of sleep: a toy dataset termed "Patches", MNIST, Fashion MNIST, CUB-200 and CIFAR-10 (Lecun et al. 1998; Xiao, Rasul, and Vollgraf 2017; Welinder et al. 2010; Krizhevsky and Hinton 2009). The Patches dataset consisted of binary patterns each belonging to its own class. The main advantage of this toy dataset is that it allows direct control over the amount of interference (number of overlapping pixels) in each of the binary patterns. This dataset was used to show the benefits of the sleep algorithm in a simpler setting and to reveal the exact weight changes which occur during sleep replay, which led to a reduction in catastrophic forgetting. To ensure generalizability of our approach, we tested the sleep algorithm on the MNIST, Fashion MNIST, CIFAR10, and CUB-200 datasets. The MNIST and Fashion MNIST datasets are widely used in machine learning, consisting of 60,000 training images of hand-written digits or fashion items and 10,000 testing images.

CIFAR-10 is a similarly sized dataset with 10 classes of low-resolution natural images ranging from airplanes to frogs. For CIFAR-10, we used extracted features from a convolutional network with 3 VGG blocks. The first block consists of 2 convolutional layers with 32 3x3 filters in each layer, followed by a max pooling layer. The second block uses 2 convolutional layers with 64 3x3 filters, followed by a max pooling layer. The last convolutional block consists of 2 convolutional layers with 128 3x3 filters, again followed by a max pooling layer and a flattening operation. To train the convolutional model, we used two dense layers with 1028 and 256 units in each layer. For extraction, we took all inputs following the 3 convolutional blocks after the flattening layer, and used these input features to perform incremental learning. The original convolutional backbone was trained on the Tiny Imagenet dataset and CIFAR-10 images were fed through this network to extract intermediate feature representations (Le and Yang 2015). The network was trained on Tiny Imagenet for 200 epochs using stochastic gradient descent with the

following parameters: momentum = 0.9, learning rate = 0.005, batch size = 100, and the categorical crossentropy loss function. CUB-200 contains natural images of 200 different bird species, with relatively few images per class. For CUB-200, following work done by (Kemker and Kanan 2017), we used the pre-trained Res-Net 50 embeddings. Here, the Res-Net 50 architecture was pre-trained on the ImageNet dataset (Kemker and Kanan 2017; He et al. 2016).

The ANN was trained sequentially on 5 groups of 2 classes for MNIST, Fashion MNIST, and CIFAR-10 and 2 groups of 100 classes for CUB200, following previous studies (van de Ven and Tolias 2018). In addition to testing the incremental MNIST and Fashion MNIST tasks (where 2 classes are learned during each task in training), we also tested the Multi-Modal MNIST task where first either MNIST or Fashion MNIST is learned and then during task 2, the other dataset is learned. This task tests the network's ability to develop representations of both datasets, digits and clothing, without catastrophic forgetting. After training on a single task, we run the sleep algorithm as described below before training on the next task.

4.3.2 Network Details

Dataset specific parameters for training ANNs are shown in Table 4.1. This table includes the network architecture used to train the tasks in a sequential fashion (number of hidden units per layer), the learning rate used for each task, number of epochs per task, number of classes per task, as well as the dropout percentage used to train the network. The accuracy of the network on the entire dataset is listed in Table 4.2 in the main paper under "Parallel Training". This denotes the accuracy of the network trained with the parameters listed in Table 4.1 but when the network has access to all training data during training time. Additionally, the "Sequential Training" row in Table 4.2 illustrates the performance of the ANN architecture when it is trained in an incremental

fashion without the use of sleep or any other methods. These metrics serve both as an upper and lower bound, respectively.

The same network architecture and training parameters were used in all comparisons. For the MNIST, Fashion MNIST, and Multi-modal MNIST tasks, we used a fully connected architecture with 2 hidden layers consisting of 1200 nodes in each layer, followed by a classification layer with 10 output neurons. The network was trained for 2 epochs per task, with mini-batch size of 100 images. For Multi-modal MNIST, the same network architecture was used, again with 10 output neurons. Thus, the same output neuron was used to represent both a digit and an article of clothing. For the CIFAR-10 dataset, we used extracted features from a convolutional network as denoted above. Extracted features were fed into a fully connected network with 2 hidden layers, with 1028 and 256 nodes in each hidden layer, respectively. These hidden layers also fed into a classification layer with 10 output neurons for each of the 10 classes in the dataset. This network was also trained for 2 epochs per task with mini-batch size of 100 images/feature vectors. For CUB-200, we used a network architecture consisting of 2 hidden layers with 350 and 300 nodes, connecting to a classification layer of 200 units, following work done by (Kemker and Kanan 2017). The network was trained for 50 epochs per task.

In all datasets, the ReLU non-linear activation function was used during awake training in all layers. Each neuron in the network was trained without a bias term, which aids in the conversion to a spiking neural network (with Heaviside-activation function) during the sleep stage (P. U. Diehl et al. 2015). The networks were trained using the basic stochastic gradient descent optimizer with momentum. Additionally, all networks were trained with the multi-class cross-entropy loss function. For incremental learning, this loss function was evaluated solely on the task being presented to the network during training time. For comparisons between different methods, such

as EWC and SI, the same training parameters and architectures were used. See Supporting Information for more details on these regularization methods used to alleviate catastrophic forgetting.

Table 4.1: Neural network training parameters

	MNIST/f. MNIST	CUB-200	CIFAR-10
Arch. size	1200,1200,10	350,300,200	1028,256,10
Learn Rate	0.065	0.1,0.01	0.1
Epochs/task	2	50	2
# class/task	2	100	2
Momentum	0.5	0.5	0.5
Dropout	0.2	0.25	0.2

4.3.3 Sleep Replay Algorithm

Here, we provide pseudocode (Figure 4.1) and more information about the sleep replay algorithm (SRA) described in the main text. The intuition behind the SRA is that a period of off-line, noisy activity may reactivate network nodes that were responsible for representing earlier tasks. If network reactivation is combined with unsupervised learning, SRA will then strengthen corresponding pathways through the network. If information about previously learned tasks is still present in the synaptic weight matrices, then SRA may be able to rescue apparently lost information.

We start in the Main procedure, where first a network is initialized, e.g., within PyTorch environment. Then, a task t is presented to the network and the network is trained, as usual, via backpropagation and stochastic gradient descent. After supervised training phase, SRA is implemented within the same environment. During the SRA phase, the network's activation function is replaced by a Heaviside function and weights are scaled by the maximum activation in each layer observed during last training. The scaling factor and layer-wide Heaviside activation thresholds are determined based on a preexisting algorithm (P. U. Diehl et al. 2015). During the

SRA phase, we start with a forward pass, when the noisy input is created and fed through the network in order to get activity (spiking behavior) of all layers. Following the forward pass, a backward pass is run to update synaptic weights. To modify network connectivity during sleep we use an unsupervised simplified Hebbian type learning rule, which is implemented as following: a weight is increased between two nodes when both pre- and post-synaptic nodes are activated (i.e., input exceeds Heaviside activation function threshold); and a weight is decreased between two nodes when the post-synaptic node is activated but the pre-synaptic node is not (in this case, another pre-synaptic node is responsible for activity in the post-synaptic node). After running multiple steps of this unsupervised training during sleep, the final weights are rescaled again (simply by removing the original scaling factor), the Heaviside-type activation function is replaced by ReLU, and testing or further supervised training on new data is performed. This all is implemented by a simple SRA function call after each new task training. Code will be made available on Github with the exact parameters dictating neuronal firing thresholds and synaptic scaling factors for each dataset and each architecture.

4.3.4 Stimulation During Sleep Phase

During sleep phase, to ensure network activity, the input layer of the network is activated with noisy binary (0/1) inputs. In each input vector (i.e., for each forward SRA pass), the probability of assigning a value of 1 (bright or spiking) to a given element (input pixel) is taken from a Poisson distribution with mean rate calculated as a mean intensity of that input element across all the inputs observed during all of the preceding training sessions. Thus, e.g., a pixel that was typically bright in all training inputs would be assigned "1" more often than a pixel with lower mean intensity. Alternatively, the mean rate of the Poisson-distribution used to create inputs may

be chosen independently on the past ANN activation which still leads to the partial recovery of the old tasks.

4.3.5 Including old data during training

In addition to testing the basic SRA when only mean input activation across all previous tasks is saved, we also tested how SRA can be complementary to existing state-of-the-art generative/rehearsal methods. To test this, we performed two additional experiments: one denoted "rehearsal" and another based off of a near state-of-the-art method, iCaRL. The "rehearsal" method included a percentage of old task data during new task learning sessions. This is a simplification of current rehearsal methods, which commonly use a separate network to generate old data, rather than storing old examples, but it still illustrates the complementary effect of utilizing both explicit "replay" during training and implicit spontaneous replay during sleep. The exact images from old tasks were randomized and the fraction of old images stored was defined by degree of rehearsal. Thus, if task 1 has 5000 images and task 2 has 5000 images, then during training with 2% rehearsal, 2% of task 1 images were stored (i.e., 100 random images and their corresponding hard-target labels were stored from task 1) and incorporated into the task 2 training dataset. When iCaRL method was tested, the fraction of stored images was defined by memory capacity, K (Rebuffi et al. 2017) and these images were chosen based on the herding exemplar method used in iCaRL. In addition, iCaRL utilizes the nearest mean of exemplar classification scheme as well as a loss function incorporating distillation on old tasks and classification on new tasks. A more detailed description of other methods tested in the paper can be found in the Supporting Information or the original papers describing these methods.

4.3.6 Analysis of Replay

For Figures 4.6 and 4.7 (main text), we defined the neurons that were specific to certain input classes. To define task-specific neurons, we presented inputs from each class for 25 forward passes in the sleep network and recorded the number of spikes (number of times each neuron exceeded its Heaviside-based activation threshold) in each neuron for each class of input. We ignored connections from the last hidden layer to the output layer in order to identify neurons that were more responsive to Task 1 or Task 2, while ignoring the actual classification component of the network. These spike counts were averaged across all input classes and sorted based on which input class maximally activated a given neuron. We defined task-specific neurons as the top 100 neurons that responded to a specific class of inputs. After task-specific neurons were labelled, we performed SRA and analyzed the change in activation input before and after sleep and firing rates during sleep of these task-specific neurons to create Figures 4.6 and 4.7.

4.4 Results

4.4.1 Sleep replay prevents catastrophic forgetting in ANNs

In animals and humans, spontaneous neuronal activity during sleep correlates with that during awake learning (Ji and Wilson 2007). This phenomenon, called sleep replay, along with Hebbian plasticity, plays a role in strengthening important and pruning irrelevant synaptic connections underlying sleep-dependent memory consolidation (Rasch and Born 2013; Tononi and Cirelli 2006). To integrate the effect of sleep into artificial neuronal systems, we interleaved incremental ANN training using backpropagation with periods of simulated sleep-like activity based on local unsupervised plasticity rules (see Methods for details). With this approach we were able to combine the "best of both worlds" - state of the art training performance delivered by modern deep neural network architectures (Kriegeskorte 2015) and important properties of

biological sleep, including local plasticity and spontaneous reactivation (Rasch and Born 2013). Importantly, our approach is fully executed within a standard machine learning environment, e.g., PyTorch or TensorFlow, and the sleep function can be easily added to any ANN type and any training algorithm.

In order to implement a sleep replay phase with local plasticity rules (referred to as Sleep Replay Algorithm (SRA) below), the network's activation function was replaced by a Heaviside function (to mimic spike-based communications that occur in the brain) and network weights were scaled by the maximum activation in each layer observed during last training, in order to increase activity during the sleep phase. The scaling factor and layer-wide Heaviside activation thresholds were determined based on preexisting algorithms developed to run trained ANNs on neuromorphic hardware, such as spiking neural networks (P. U. Diehl et al. 2015). To modify network connectivity during sleep phase we used an unsupervised, simplified Hebbian type learning rule, which was implemented as follows: synaptic weights between two neurons are increased when both pre- and post-synaptic neurons are activated sequentially; synaptic weights are decreased between two neurons when the post-synaptic node is activated but the pre-synaptic node is silent (does not reach activation threshold). Further, to ensure sufficient network activity during the sleep phase, the input layer of the network was activated with noisy binary inputs. In each input vector at each time step of SRA, the probability of assigning a value of 1 (bright or spiking) to a given input pixel is taken from a Poisson distribution with mean rate calculated as the mean intensity of that input element across all the inputs observed during all of the preceding training sessions. Thus, e.g., a pixel that was typically bright in all training inputs would be assigned "1" more often than a pixel with lower mean intensity. Therefore, the only old task information that needs to be stored for future SRA applications is the mean input layer activation across all the past tasks and this

information does not scale with the number of tasks. Importantly, no inputs representing specific memories were ever presented to the network during sleep; the state of the network (weight matrices) implicitly determined the patterns of reactivation and, ultimately, what was replayed during sleep.

In this study we analyzed SRA in the context of both class-incremental learning and cross-modal tasks. Class-incremental learning occurs when a network learns a series of classes (e.g. MNIST digits) incrementally without access to previously learned classes. In this case, performance is measured as the network's ability to classify and distinguish all classes. Cross-modal tasks measure the network's ability to store two distinct tasks (e.g., MNIST digits and Fashion MNIST images) in the same parameter space. We first utilized a "toy" example of binary patterns to analyze how synaptic weights change during sleep to support incremental learning. We then tested SRA in an incremental learning framework on the MNIST, Fashion MNIST, CUB-200 and CIFAR10 datasets (see Methods). For cross-modal tasks, we measured the ability of an ANN to learn sequentially both the MNIST and Fashion MNIST datasets when the network could only access one dataset during training time.

4.4.2 SRA promotes consolidation of overlapping binary patches

We first tested the sleep algorithm by training a small network with just an input and output layer to distinguish four binary 10x10 images (see Fig. 4.2A) presented sequentially as two tasks (Task 1 - first two images; Task 2 - second two images). The network was always tested on its ability to classify all 4 images using a softmax classifier. The amount of interference was measured as the number of overlapping pixels between images. Catastrophic forgetting should not occur when there is no interference between the images but, as the number of overlapping pixels increases, new task training can lead to forgetting. Our studies using biophysical models of the

thalamocortical network (Gonzalez et al. 2019; Golden et al. 2020) revealed that catastrophic forgetting occurs because the network connectivity becomes dominated by the most recently learned task, so input patterns for old tasks are insufficient to activate their respective output neurons; sleep replay can "re-balance" the network to allow correct recall of the old memories.

After training on the first two images (denoted T1), the network could classify T1 images accurately but has not yet learned the other two images, so overall performance was 50% (Fig. 4.2B, dashed blue line). Here, when, e.g., the first image was presented to the network, input to its corresponding output neuron was greater than the maximum input to the other output neurons (Fig. 4.2C, left group of bars, 12 pixel overlap) - correct classification. After training on the second two images (denoted T2), the network either learned them without interference to T1 (performance increased to 100% - all 4 images are classified correctly), when there was little overlap between tasks (Fig. 4.1B, dashed red line, overlap is less than 10 pixels), or forgot the first two images (performance remained at 50% - T1 is erased and everything is classified as T2), when there was large overlap between tasks (Fig. 4.2B, dashed red line, overlap is more than 10 pixels). In the last case, presenting the first image resulted in greater activation of T2 output neurons (Fig. 4.2 C, red bar in middle group) than the first output neuron (Fig. 4.2C, black bar in middle group) - catastrophic forgetting. When SRA was applied following T2 (Fig. 4.2B, yellow line), T1 was recovered even for large overlaps between tasks (compare yellow and red lines for overlaps more than 10 pixels in Fig. 4.2B). Here, we observed that although input to the first class output neuron (upon first image presentation) remained unchanged (Fig. 4.2C, black bar in rightmost group), the input to the other output neurons decreased following SRA (Fig. 4.2C, red bar in rightmost group). Thus, after SRA, the network was able to withstand larger amounts of interference, indicating that SRA is beneficial in reducing catastrophic interference for this simplified task. It is worth

mentioning that after training T2, the input to correct T1 output neuron was reduced but did not become negative (Fig. 4.2C, black bar in middle group). Thus, T1 images would still activate T1 output neurons, but not as strongly as they activate T2 output neurons - misclassification but not complete forgetting of the T1 weight structure, so it still can be resurrected by SRA.

We next examined the synaptic weight changes after training the second task, and after SRA. Figure 4.2D shows histograms of synaptic weights from input layer neurons to four output neurons (T1 neurons on the left and T2 neurons on the right) for the 12-pixel overlap condition, which results in catastrophic forgetting following T2 training and complete recovery following SRA. We separated weights from input neurons representing uniquely T1 and T2 pixels (Figure 4.2D, green, orange), as well as overlapping pixels between T1 and T2 (purple). After T1 training (Figure 4.2D, top row), the weights from T1 pixels (both unique and overlapping) to T1 decision neurons increased (purple and green distributions on the left), while connections to T2 decision neurons became negative (purple and green on the right). After T2 training (Figure 4.2D, middle row), the weights from T2 pixels (both unique and overlapping) to T1 output neurons decreased (purple and orange on the left), while connection to T2 output neurons increased and became positive (purple and orange on the right). Strong input to T2 output neurons from T1 overlapping pixels (purple on the right) overcame input to T1 output neurons from unique T1 pixels (green on the left, see also Fig. 4.2C, middle group). This resulted in catastrophic forgetting of T1, as T1 inputs led to higher activation of T2 output neurons. After SRA (Figure 4.2D, bottom), weights from T1 unique pixels to T2 output neurons became inhibitory (green distribution on the right) while most other categories of weights remained unchanged. Thus, before SRA, presenting T1 images resulted in preferential activation of T2 output neurons and misclassification. After SRA

the same T1 inputs inhibited T2 output neurons, so T1 output neurons displayed relatively higher activation leading to correct classification.

Since T1 neurons did not spike during sleep in this simple model, the weights to T1 output neurons did not change (compare Figure 4.2D left, middle vs bottom row); however, for more complex tasks and network architectures, all the weights could change after sleep phase. This simple model analysis revealed that SRA down-scales synaptic weights from task-irrelevant neurons, thereby reducing "cross-talk" between tasks. A more rigorous analysis of this toy model revealed that following training of T1-T2 the network weights have positive cosine similarity to the weights of the network trained on T1 alone (See Supporting Information, Section 1). This demonstrates that even when catastrophic forgetting is observed from classification perspective, the network weights preserve information about previous tasks.

4.4.3 SRA reduces catastrophic forgetting on standard datasets

ANNs have been shown to suffer from catastrophic forgetting for various standard image datasets including MNIST, CUB-200 and CIFAR-10 (Kemker and Kanan 2018). To test SRA for these datasets, we created 5 tasks (per dataset) for the MNIST, Fashion MNIST, and CIFAR-10 datasets and 2 tasks for the CUB200 dataset. Each pair of items in the MNIST (e.g., digits 0 and 1), Fashion MNIST and CIFAR-10 datasets was defined as a single task, and half of the classes in CUB200 was considered a single task. Tasks were trained sequentially, and each new task training was followed by a sleep phase until all tasks were trained. This mimics interleaving periods of awake training with periods of sleep in the mammalian brain.

A baseline ANN with two hidden layers (see Methods for details) trained incrementally without sleep suffered from catastrophic forgetting, representing the lower bound on performance (Table 4.2, Sequential Training). The ideal accuracy of the same network trained on all tasks at

once represents the upper bound (Table 4.2, Parallel Training). We found a significant improvement in the overall performance compared to the lower bound (Table 4.2, SRA vs Sequential Training), as well as task specific performance (Figure 4.3) when SRA was incorporated into the training cycle. On CUB-200, the baseline ANN suffered from catastrophic forgetting after it was trained sequentially on two tasks (first task - 5%, second task - 95%). Incorporating SRA after each task training resulted in much higher and balanced classification accuracy (first task - 63.2%, second task - 45.4%). Similar results were found for CIFAR-10, where the network with SRA achieved overall accuracy values of 35%, significantly higher than the control ANN without SRA (~19%). Errors in Table 4.2 represent the standard deviation across 5 trials with different network initialization and different task orders. Note that computational costs for running SRA are comparable with the costs of training each additional task (when task training is implemented in batches). However, sleep required much less inputs to pass through the network; thus, the computational performance of sleep phase can be likely improved by incorporating the idea of mini-batches (see Supporting Information, Figure 4.9 for more detail).

We next tested the SRA using a cross-modal task, where the network first learned the MNIST and next the Fashion MNIST dataset. An ideal network trained on both datasets at once achieved a classification accuracy of around 90%. When trained incrementally, the baseline ANN failed to accurately classify the MNIST data, achieving overall classification accuracy of 47% (Table 4.2) Incorporating SRA into the training boosted overall classification accuracy to 61%. While we primarily tested a network with only two hidden layers, the analysis of 4-hidden layer networks on MNIST task revealed that SRA can be applied in deeper architectures to recover the same level of performance.

Table 4.2: Average test accuracy (± s.d) (averaged across different task orders) for baseline sequential training, Elastic Weight Consolidation, Synaptic Intelligence, Orthogonal Weight Modification, Sleep Replay Algorithm, Rehearsal with 2% of old data stored, Sleep Replay Algorithm + Rehearsal (2% of old data stored), and the ideal performance of a network trained on all data at once.

Method	Inc. MNIST	Inc. Fashion MNIST	Multi-modal MNIST	Inc. CUB-200	Inc. CIFAR10
Sequential Training	19.26 ± 0.0001	19.96 ± 0.0001	47.18 ± 0.0020	5.32, 95.41	18.81 ± 0.0005
EWC	20.03 ± 0.77	20.18 ± 0.29	74.55 ± 0.83	0.0, 63.85	18.13 ± 0.0012
SI	22.02 ± 1.70	22.05 ± 2.05	74.15 ± 0.80	0.07, 60.01	19.8 ± 0.0049
OWM	56.32 ± 2.70	49.33 ± 5.74	91.29 ± 1.05	71.4, 21.5	44.78 ± 1.7957
SRA	44.87 ± 0.0630	42.03 ± 0.0175	61.33 ± 0.0150	63.2, 45.4	34.56 ± 0.0127
Rehearsal (2\% data stored)	46.5 ± 2.987	51.81 ± 0.767	86.48 ± 0.253	49.21, 51.26	30.86 ± 0.726
Rehearsal + SRA	64.812 ± 2.91	56.196 ± 5.64	86.74 ± 0.367	62.14, 34.93	41.58 ± 0.974
Parallel Training	94.19 ± 0.0011	83.48 ± 0.003	90.05 ± 0.0028	85.49, 79.15	67.36 ± 0.0017

Although we report here SRA performance numbers lower than those for some generative models (van de Ven, Siegelmann, and Tolias 2020a; Kemker and Kanan 2017), SRA was able to reduce catastrophic forgetting with only limited (average statistics) knowledge of previously learned examples and solely by utilizing spontaneous replay driven by the weights important for representation of old tasks. Among methods operating without access to old data, SRA surpassed regularization methods, such as Elastic Weight Consolidation (Table 4.2, EWC) and Synaptic Intelligence (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017), on all incremental learning tasks (Table 4.2, SI) and revealed reduced performance only for multi-modal task (see Discussion and Supporting Information). However, a recently developed regularization method (Orthogonal Weight Modification, OWM) slightly surpasses performance of SRA on most tasks, suggesting that regularization methods can promote recovery of old tasks in a class-incremental learning

setting (Zeng et al. 2019). One implementation advantage of SRA over OWM is that SRA is an offline method, so it can be run after the normal training process is completed. Therefore, it can be complimentary to any other continual learning method. Furthermore, for tasks where it is unknown a priori if/when new training would be needed, SRA can be easily applied post fact, as only past average input would be needed to run SRA. In contrast, with OWM, one will need to feed all previous inputs through the network to compute projections.

Ultimately, our results suggest that information about old tasks is not completely lost even when catastrophic forgetting is observed from the performance-level perspective. Instead, information about old tasks remains present in the synaptic weights and can be resurrected by off-line processing, such as sleep replay.

4.4.4 SRA is complementary to state-of-the-art rehearsal methods

Many generative solutions aimed at solving catastrophic forgetting train a separate generator network to recreate and make use of (a fraction of) the old data during new training sessions - commonly called “replay” - to prevent forgetting (van de Ven, Siegelmann, and Tolias 2020a; Kemker and Kanan 2017). We next tested the complementary effect of incorporating old training data during new training sessions (“replay”) along with SRA. In this scenario, we included a small percentage of old task data during new task training (see Methods and Supporting Information). The small amount of old task examples was alone (without SRA) insufficient to substantially increase the network’s classification accuracy on most tasks (Table 1, Rehearsal, Figure 4.4, gray lines). When SRA was included, this boosted the overall classification accuracy when compared to using a small fraction of old data alone (Table 1, Rehearsal+SRA vs Rehearsal, Figure 4.4) These results suggest that SRA can reduce the amount of data needed to be generated or stored with state-of-the-art rehearsal methods, while still obtaining near ideal accuracy. Note

that the training time used in Table 4.2 was based on determining when a sequentially trained network reached its optimal performance on a single task. We, however, discovered that rehearsal methods and OWM may need longer training time in order to reach their optimal performance (see Supporting Information and Table 4.1). Thus, we hypothesize that SRA could support these rehearsal methods by reducing the training time in addition to reducing memory capacity requirements. This point is explored below.

From a neuroscience standpoint, this result predicts that learning new memories first by a separate (such as hippocampal) network, that would be storing new memories but also some high-level information about old memories ("indexes" (Teyler and DiScenna 1986)), and subsequently training another (such as cortical) network using output of that hippocampal network, followed by cortical replay, gives an optimal solution to embed new memories in cortex and protect old memories. This is indeed how we believe brain learns hippocampus dependent declarative memory tasks (Rasch 2013).

In this section, we show how incorporating SRA on top of a specific rehearsal method (iCaRL) can result in better continual learning performance than iCaRL alone (details on iCaRL can be found in Supporting Information or the original paper (Rebuffi et al. 2017)). Table 4.3 shows performance when different memory capacities (K, number of examples from previous classes that can be stored) are used. With a memory capacity of K and a network trained on t classes, m =K/t examples are stored from each class. The first two rows show a comparison of SRA vs. iCarl with K=100. SRA alone has equivalent memory capacity of K=1 since it stores only average input. We found that SRA alone showed significantly higher performance compare to iCaRL with K=100. Moving to K=1000, SRA was able to substantially improve the performance of iCaRL, by 10-40% on the different datasets, when combined together. Finally, moving to

K=20000, both iCaRL alone and iCaRL+SRA approached the accuracy of an ideal network, with iCaRL+SRA sometimes exceeding this (in the case of MNIST). In sum, SRA could support various rehearsal methods by a) reducing the amount of old data that are stored, or b) allowing only replay of the highest-quality generated examples. In all cases, this would reduce training time and alleviate catastrophic forgetting, while needing fewer data from old datasets during new training.

Next, we discuss in more details how incorporating SRA on top of a state-of-the-art rehearsal method, iCaRL, can result in better continual learning performance than iCaRL alone (details on iCaRL implementation can be found in Supporting Information or the original paper (Rebuffi et al. 2017). Table 4.3 shows performance when different memory capacities (K, number of stored examples from previous classes) were used. In the case of MNIST, iCaRL with K = 100 achieved a performance of 65.502% after 10 epochs/task and iCaRL with K = 200 achieved a performance of 76.856% after 10 epochs/task. iCaRL+SRA with K = 100 achieved a classification performance of 78.086% after 10 epochs/task. Thus, a higher accuracy can be obtained with iCaRL + SRA and it may even be achieved with a lower memory capacity. For MNIST, Fashion MNIST and CIFAR10 datasets, in almost all cases (except K = 2000 for MNIST) iCaRL + SRA had higher performance than iCaRL alone for the same memory capacity. We also observed that OWM could benefit from longer training times. However, the performance of iCaRL + SRA with K = 100 always exceeded OWM performance, suggesting that rehearsal methods are still the state-of-the-art in class-incremental continual learning settings.

Note that these experiments were run for a larger number of epochs/task as compared to Table 4.2. Thus, we hypothesized that in addition to lowering memory requirements, SRA could also reduce training time (denoted as number of epochs per task) needed to achieve optimal results.

Indeed, we found that iCaRL + SRA converges more rapidly than iCaRL alone. For example, for K = 50, iCaRL + SRA achieved after just one epoch/task of training the same accuracy as iCaRL alone after 10 training epoch/tasks. For K = 100, 4 epochs/task were required with iCaRL + SRA before the same 10 epoch/task accuracy was obtained with iCaRL alone. For K = 1000, iCaRL + SRA after 8 epochs/task had a final accuracy of 87.698%, whereas iCaRL even after 10 epochs/task only achieved a final accuracy of 87.32%. In general, the benefits of SRA were higher for lower values of K. We defined the training savings as the number of epochs/task after which iCaRL + SRA achieves a greater performance than iCaRL alone after 10 epochs/task. The training savings on all 3 datasets (averaged across all memory capacities and task orders) were: 3.73 epochs/task for MNIST, 3.67/task epochs for Fashion MNIST, and 2.80 epochs/task for CIFAR10 (see Supporting Information for example plots).

In sum, we found that SRA can support various rehearsal methods by a) reducing the amount of old data that are stored (or allowing only replay of the highest-quality generated examples); b) reducing training time.

Table 4.3: SRA improves upon iCaRL method with different memory capacities

	K = 50	K = 100	K = 200	K = 500	K = 1000	K = 2000
iCaRL, MNIST	53.428 ± 5.20	65.502 ± 4.66	76.856 ± 5.29	87.326 ± 1.30	90.628 ± 0.41	92.850 ± 0.44
SRA + iCaRL, MNIST	69.97 ± 3.74	78.086 ± 3.16	84.498 ± 1.39	88.862 ± 0.44	91.130 ± 0.58	92.742 ± 0.41
iCaRL, Fashion MNIST	49.342 ± 6.83	57.786 ± 3.13	62.828 ± 2.97	69.038 ± 2.78	73.972 ± 1.58	78.030 ± 0.62
SRA + iCaRL, Fashion MNIST	51.554 ± 11.63	61.916 ± 5.25	65.110 ± 2.95	69.798 ± 2.48	75.226 ± 1.28	78.542 ± 1.17
iCaRL, CIFAR10	35.156 ± 3.41	43.244 ± 1.99	49.102 ± 2.07	54.898 ± 1.49	59.528 ± 0.77	62.878 ± 0.65
SRA + iCaRL, CIFAR10	39.382 ± 3.61	46.002 ± 2.07	51.324 ± 1.86	57.504 ± 0.61	61.23 ± 0.88	64.018 ± 0.56
			MNIST	Fashion MNIST	CIFAR10	
		Sequential Training	19.49 ± 0.002	19.67 ± 0.003	19.01 ± 0.002	
		Parallel Training	98.02 ± 0.006	87.86 ± 0.005	72.43 ± 0.002	
		OWM, 10 epochs/task	77.038 ± 2.91	58.35 ± 2.05	34.234 ± 1.87	

4.4.5 SRA reduces catastrophic forgetting by replaying old task activity

How does SRA work? From a neuroscience perspective, sleep reduces interference by replaying activity of recently learned tasks and old relevant (interfering) tasks (Rasch and Born 2013). Using biophysical models of brain network and testing for simplified task of learning

overlapping memory sequences, we showed that sleep replay modifies the synaptic weights to create unique synaptic representation for each task (Gonzalez et al. 2019). Such differential allocation leads to reduced representational overlap and therefore diminishes catastrophic forgetting.

To address how SRA works for ANNs, we examined a reduced class-incremental learning MNIST task. The network was first trained on digits 0 and 1 (Task 1), followed by sleep. Next, the network was trained on digits 2 and 3 (Task 2), leading to T1 forgetting, followed by second sleep phase. We then implemented the second sleep phase after T2 training and we then implemented and analyzed the second period of sleep that resulted in recovery of performance on the first task (overall accuracy on both tasks after SRA = 90%).

To test if after SRA the neurons are differentially allocated to represent digits from different tasks, we first looked at the correlation of activities in the hidden layers. The ANN was presented with all inputs from the test sets of both tasks and we calculated the average correlation within the first and second hidden layers before and after application of SRA (Figure 4.5). In both layers, before SRA (when all inputs were classified as either 2 or 3), correlations of activity for digits from different classes were almost as high as correlations of activity for digits from the same class (with exception of classes 2 and 3 in the 2nd hidden layer) (Figure 4.5B, left). After SRA we observed decorrelation (near-zero correlations between representation of digits from different classes) and an increase in representational sparseness (where each stimulus strongly activates only a small subset of neurons) for all four digits (Figure 4.5, right and Figure 4.14). This suggests that SRA prevents interference between classes by allocating different neurons to different tasks, thereby creating a distinct population code for different input classes.

Decorrelation of activity alone may not explain the old task recovery. Indeed, if activity of neurons representing the earlier task remains lower than that for the later task, then catastrophic forgetting would still occur. Therefore, we next analyzed activation properties of the neurons representing individual tasks (Figure 4.6) In each hidden layer, we selected the top 100 neurons that were most active in response to Task 1 or Task 2 inputs (see Methods for details on selection process) and analyzed how input to these neurons changes after SRA. In the first hidden layer, both Task 1- and Task 2-neurons experienced a decrease in input strength, but the effect was generally higher for Task 2 neurons, reflecting a greater decrease in the weights connecting to Task 2-neurons (Figure 4.6, left). More notably, in the second hidden layer, we observed an increase of the input to Task 1-neurons but a decrease of the input to the Task 2-neurons (Figure 4.6, right). This suggests that a relative increase in activity of Task 1-neurons along with an overall decorrelation of representations between the tasks explains recovery of performance on the old task.

In the mammalian brain, sleep dependent memory consolidation occurs through memory replay, i.e., patterns of neurons activated during task learning are reactivated spontaneously during sleep (Ji and Wilson 2007). To test if replay happens during SRA, we looked at the firing activity of Task 1- and Task 2- neurons during sleep. To avoid possible bias from using task averaged input, here the network was stimulated during sleep by a completely random input. We calculated the average firing rates of neurons and we found in the first hidden layer that the top 100 most active neurons involved in representation of the previously learned individual digits had a higher average firing rate during sleep than a randomly selected subset of neurons (Figure 4.7A). In the second layer, this result was more pronounced for the most recently learned task (digits 2 and 3) (Figure 4.7B, Figure 4.13). To compare firing rates of digit-specific neurons, we concatenated the

firing rates of digit-2 and digit-3 (Task 2) neurons and compared them with the concatenated firing rates of digit-0 and digit-1 (Task 1) neurons. We found that digit-2 neurons and digit-3 neurons were significantly more active in layer one ($t(200) = 3.456$, $p = 0.004$, 1-sided t-test, Bonferroni Correction) and layer two ($t(200) = 5.215$, $p < 0.001$, 1-sided t-test, Bonferroni Correction). This suggests that spontaneous firing patterns during sleep are correlated with activity observed during task learning, in agreement with neuroscience data (Ji and Wilson 2007; Rasch and Born 2013). Interestingly, sleep replay improved performance not only by increasing connectivity for the old tasks but also by reducing connectivity representing the most recent task.

If replay indeed provides a mechanism of how sleep protects memories from interference, then for a single task, sleep replay should increase performance. Indeed, the most well-established in neuroscience effect of sleep on memory is either mapping memory traces from hippocampus to the cortex for declarative memory tasks or augmenting cortical traces for procedural memory tasks (Rasch and Born 2013). In this new study we typically observed some reduction in performance on most recent task. Do we have contradiction here? To test this, we tested effect of SRA on a single task memory performance as a function of the amount of initial training. We found (see Figure 4.12) that when the network is undertrained, i.e., initial performance is low, SRA can greatly increase performance without involving any new training data. However, if memory is well trained, SRA cannot improve or even slightly reduces performance.

4.5 Discussion

We implemented an unsupervised sleep replay algorithm (SRA) for artificial neural networks and we showed that SRA can alleviate catastrophic forgetting for several different datasets, ranging from binary patterns to natural images. We found that SRA simulates properties of biological sleep replay: (a) spontaneous activation of neurons during sleep leads to replay of

previously learned activation patterns, (b) unsupervised Hebbian type plasticity modifies the connectivity matrix to increase task specific connections and to prune excessive connections between neurons. This increases sparseness of representations and reduces representational overlap amongst distinct input classes. Our results suggest that unsupervised Hebbian type plasticity combined with spontaneous activity during sleep can help alleviate catastrophic forgetting in an incremental learning setting.

Existing approaches to prevent catastrophic forgetting generally fall under two categories: rehearsal and regularization methods (Kemker et al. 2018). Rehearsal methods combine previously learned data, either stored or generated, with novel inputs in the next training to avoid forgetting (Hayes, Cahill, and Kanan 2019; Robins 1995; van de Ven, Siegelmann, and Tolias 2020b; Shin et al. 2017). This approach includes models where distinct networks, loosely representing hippocampus and cortex, are used to generate examples from a distribution of the previously learned tasks (Kemker and Kanan 2017). As the number of previously learned tasks grows, this approach would require increasingly complex generative networks capable of potentially generating everything that was learned before. In contrast, brain networks are capable of spontaneously replaying previously learned memories during sleep simply by reactivating previously learned synaptic weights patterns. Thus, although rehearsal methods work well from an engineering standpoint, they unlikely capture the mechanisms that nature developed to enable continual learning. While current SRA performance, as we report here, is somewhat inferior compared to the state-of-the-art rehearsal techniques (van de Ven, Siegelmann, and Tolias 2020b; Hayes, Cahill, and Kanan 2019; Kemker and Kanan 2017), our approach does not require storing any task specific information or training new generator networks. SRA may be complementary to rehearsal methods by incorporating partial old data replay, reducing the amount of old data needed

for new task training. Importantly, SRA was also shown to perform other memory functions associated with sleep, such as promoting generalization and improving robustness against adversarial attacks (Tadros et al. 2019) . While actual training data were used for sleep replay in (Tadros et al. 2019), here we show that sleep replay can alleviate catastrophic forgetting on class incremental learning task by only having access to the basic input statistics.

Regularization approaches (Li and Hoiem 2017) for reducing catastrophic forgetting aim to modify plasticity rules by incorporating additional constraints on gradient descent such that important weights from previously trained tasks are maintained. The Elastic Weight Consolidation (EWC) and Synaptic Intelligence (SI) methods penalize updates to weights deemed important for previous tasks (Kirkpatrick et al. 2017; Zenke, Poole, and Ganguli 2017). Although these studies report positive results in preventing catastrophic forgetting in various tasks (MNIST permutation task, Split MNIST, Atari games), they may not work well in a class-incremental learning framework, where one class of inputs is learned at a time (van de Ven, Siegelmann, and Tolias 2020a). Since these approaches support continual learning by stabilizing weights that are deemed important for earlier tasks, high overlap in representation between tasks belonging to the same dataset may reduce their performance (see Supporting Information). In terms of memory constraints, SRA is similar to regularization methods since only a fixed amount of information, independent of the number of tasks trained, is needed to be stored, but SRA exceeds performanace of EWC and SI approaches in an incremental learning framework that is natural for biological systems. Orthogonal Weight Modification (OWM) improves upon these methods by computing a projector matrix for old tasks and only allowing weight updates in an orthogonal direction to this projector (Zeng et al. 2019). OWM was shown to be more successful than EWC and SI on class-incremental learning tasks.

Our approach is implemented using a machine learning framework, where during the sleep phase we simply substitute in the Heaviside activation function and modify the learning rule. Since the idea of SRA is inspired by biological spiking networks (SNNs), we also tested SNN-only implementation by performing ”awake” training on MNIST data in an SNN via backpropagation (Yanguas-Gil 2020) and found that incremental learning can lead to catastrophic forgetting. When sleep phase was implemented after each new task training using noise driven spontaneous replay, this recovered the old task performance (see Supporting Information). It remains open question if SRA would benefit convolution layers of the network, which were kept frozen in this study. Recent studies suggest that catastrophic forgetting does not occur in the feature extractor but takes place in the later layers, which motivated the use of SRA in these layers (Ramasesh, Dyer, and Raghu 2020; Goodfellow et al. 2013). From neuroscience perspective convolutional layers are somewhat equivalent to the primary visual cortex which is less plastic beyond development and unlikely involved in significant rewiring during sleep replay, in contrast, e.g., to the associative cortices.

Sleep replay helps to resurrect performance on tasks that were damaged after new training. This suggests that while ANN performance for old tasks is reduced, the network connectivity retains partial information about these tasks and spontaneous activity combined with unsupervised plasticity during sleep may reverse damage and reorganize connectivity to accommodate both tasks. Previously, it was shown that a wake-sleep algorithm developed for recurrent spiking neural networks, which does capture some principles of memory consolidation during sleep, can reduce the number of training examples needed to achieve optimal performance on single tasks (Thiele, Diehl, and Cook 2017). The wake-sleep algorithm for Boltzmann Machines was shown to be able to learn representations of inputs and highlighted the role of incorporating a sleep phase to improve

learning (Hinton et al. 1995). Our work further adds to this literature by exploring the role of sleep and local plasticity rules in overcoming catastrophic forgetting.

Our study makes several predictions for neuroscience: (a) Synaptic dynamics during sleep remain poorly understood. Some studies suggest net reductions of synaptic weights (Tononi and Cirelli 2006), while others argue for net increase (Igor Timofeev and Chauvette 2017). Our work (extending ideas of (Gonzalez et al. 2019)) predicts that sleep replay leads to complex reorganization of synaptic connectivity, including potentiation of some synapses and pruning of others with the goal of increasing separation between memories. We found that sleep replay may increase the "contrast" between memory traces by enhancing lateral inhibition, such that activation of one memory inhibits other "similar" memories to avoid interference (Lee et al. 1999). Recent studies suggest that local learning rules can orthogonalize memories with different temporal contexts (Bouchacourt et al. 2020; Saxe, Nelli, and Summerfield 2021). Our current (and recent (Gonzalez et al. 2019)) work agrees with this idea and further suggests that local learning rules and sleep can orthogonalize representation of interfering memories, even with limited contextual information. (b) Sleep increases the sparseness of memory representations, which is in line with previous theoretical ideas about the role of interleaved training (McClelland, McNaughton, and O'Reilly 1995). (c) Currently, sleep replay ideas are best developed for NREM sleep when neuronal activity is structured by oscillations. Our model predicts that sleep replay can be mediated by sparse patterns of excitation propagating through the network, which is typical for REM sleep (indeed, memory replay was also reported in REM sleep (Louie and Wilson 2001)). (d) Our model predicts the importance of neuromodulatory changes from learning (wake) to consolidation (sleep) phase, including strengthening of intracortical synapses and reduction of intrinsic excitability, as found in NREM sleep (Gil, Connors, and Amitai 1997; McCormick 1992). We suggest that these

changes effectively increase the "signal-to-noise" ratio in network dynamics to promote the stronger memories, that are replayed and consolidated, while allowing forgetting of the weaker memory traces.

Catastrophic forgetting may be interpreted as asymmetry of the weights' configuration that became biased towards the most recent task after new training. From this perspective, SRA may be seen as a “symmetry-correcting” mechanism. Indeed, Hebbian learning has been shown to orthogonalize neural codes and our recent biophysical modelling work (Gonzalez et al. 2019), as well as results here, have shown that sleep orthogonalizes memories by “assigning” distinct weights to represent distinct memories. We found that rather than engage completion between memories, sleep allows chunks of competing memories to replay simultaneously, so each memory would reach its optimal representation, which can be seen as a way of recovering the symmetry of weight distribution across tasks.

In sum, in this study we proposed an unsupervised sleep replay algorithm, inspired by the known role of biological sleep, to recover synaptic connectivity that otherwise would be forgotten after new training in ANNs. We tested a simplified model of sleep (noisy replay) and a Hebbian-type plasticity rule. Future work may need to explore more complex patterns of sleep activity (e.g., sleep waves) and learning rules (e.g., (Gerstner et al. 2014)), which could further improve performance.

4.6 Acknowledgements

This work was supported by the Lifelong Learning Machines program from DARPA/MTO (HR0011-18-2-0021) and ONR (MURI: N00014-16-1-2829). Chapter 4, in part, is a reprint of the material as it was submitted for publication. Tadros, Timothy; Krishnan, Giri; Ramyaa, Ramyaa;

Bazhenov, Maxim. The dissertation author was one of the primary investigators and first author on this paper.

4.7 Supplementary Information

4.7.1 Analysis of catastrophic forgetting and sleep in toy model

In this section, we examine the cause of the catastrophic failure and the role of sleep in recovering from the forgetting in toy model. While this example is not intended to model all scenarios of catastrophic forgetting, it provides the intuition and explains how our algorithm prevents catastrophic forgetting. Let us consider the 3-layer network trained on two categories, each with just one example. Consider 2 binary vectors (Category 1 and Category 2) with some region of overlap.

We consider ReLU activation since it was used in the rest of this work. We assume the output to be the neuron with the highest activation in the output layer. Let the network be trained on Category 1 with backpropagation using static learning rate. Following this, we trained the network on Category 2 using same approach. A 3-layer network we consider here has an input layer with 10 neurons, 30 hidden neuron and an output layer with 2 neurons for the 2 categories. Inputs are 10 bits long with 5 bit overlap. We trained with learning rate of 0.1 for 4 epochs.

We can divide the hidden neurons into four types based on their activation for the two categories: A - those neurons that fire for Category 1 but not 2; B - those neurons that fire for Category 2 but not 1; C - those neurons that fire for Category 1 and 2; D - those that fire for neither category, where firing indicates a non-zero activation. Note that these sets may change during training or sleep. Let X_i be the weights from type X to output i.

Consider the case where the input of Category 1 is presented. The only hidden layer neurons that fire are A and C. Output neuron 1 will get the net value $A*A_1 + C*C_1$ and output

neuron 2 will get the net value $A*A_2 + C*C_2$. For output neuron 1 to fire, we need two conditions to be held: (1) $A*A_1 + C*C_1 > 0$, (2) $A*A_1 + C*C_1 > A*A_2 + C*C_2$. The second condition above can be rewritten as $A*A_2 - A*A_1 < C*C_1 - C*C_2$, which separates the weights according to the hidden neurons. Using this separation, we give the following definitions:

Define a to be $(A_2-A_1)*A$ on pattern 1; b to be $(A_2-A_1)*A$ on pattern 2; p to be $(C_1-C_2)*C$ on pattern 1 and q to be $(C_1-C_2)*C$ on pattern 2. (Note that p and q are very closely correlated since they differ only in the activation values of C neurons which are positive in both cases).

So, on the input pattern 1, output 1 fires only if $a < p$; on input pattern 2, output 2 fires only if $q < b$.

Following training on 2 categories, if the network can not recall Category 1, i.e., output neuron 1 activation is negative or less than that of output neuron 2, catastrophic forgetting has occurred (We confirmed this occurred 78% of times for the 3 layer network described above and 100 trials). The second phase of training ensures $q < b$. This could involve reduction in q which would reduce p as well. (Since A does not fire on input pattern 2, back-propagation does not alter a). Reducing p may result in failing the condition $a < p$, i.e., misclassifying input 1.

Sleep can increase difference among weights (which are different enough to begin with) as was shown in (Wei, Krishnan, and Bazhenov 2016; Gonzalez et al. 2019). So, as the difference between A_2 and A_1 increases, this decreases a (as A_1 is higher, $a = A_2 - A_1$ decreases). Occurrence of the same change to p is prevented as follows: it is likely that at least one of the weights coming to a C neuron is negative. In that case, increasing the difference would involve making the negative weight even more negative, resulting in the neuron joining either A or B (as it no longer fires for the pattern showing the negative weight), thus reducing p. (This is explained further in the supplement). When the neurons in C remains, we have a more complex case: here, a decreases,

but p may also decrease correspondingly; another undesirable scenario is when b decreases to become less than q. Typically sleep tends to drive synaptic weights of the opposite signs, or the weights of same sign but different by some threshold value, away from each other. There are conditions when the difference between weights is below threshold point to cause divergence. In those cases sleep does not improve performance.

In our experiments, for majority of the cases, we found C to be empty after sleep, thus making p to become 0. For the instances when this was not a case, the initial values of A_1, A_2, B_1 and B_2 were almost 0, i.e., the entire work of classifying the inputs is done by shared input. In such case, the network has no hidden information that sleep could retrieve (see Figure 4.8).

4.7.2 Computational costs of SRA

We can evaluate computational costs of SRA as combination of (a) number of images passed through the network (both forward and backward passes), and (b) storage costs. We first note that length of sleep is mostly related to the size of the network, not to the size of the training set. Thus, for larger datasets, while more images may be needed during training phase (in the "awake" state) in order for classification accuracy/loss to saturate, during "sleep" this is not necessarily true. Because of this, the computational cost of sleep is generally similar for networks trained on Imagenet, CIFAR, MNIST, etc (as long as the network size that sleep is applied to is the same, e.g., 2 hidden layers with the same number of units).

As for number of iterations of sleep, we have observed that two parameters dictate how much sleep is necessary to recover performance on old tasks: the magnitude of weight changes (determines how much to increase/decrease weights when an STDP event occurs) and the input firing rate (that is the maximum firing rate for neurons in the first layer during sleep). Parameters can be set so that a very small amount of sleep is needed to recover old task performance. In the

(Fig 4.9) one can see that as the network learns all 5 MNIST tasks, only ~500 iterations (and thus ~500 feedforward passes through the network) of sleep are needed to reach performance saturation. Each line shows the accuracy as a function of sleep duration on the entire dataset (all 5 tasks). Color indicates the sleep phase (e.g., after training task 1, task 2. etc.)

If sleep requires 500 iterations to ultimately converge, this is equivalent to presenting 500 "noisy" images. In contrast, during training, if we train each class for 2 epochs (as in the case with MNIST), 10,000 images are presented twice, i.e., 20,000 images fed through the network both forwards and backwards for a total of 40,000 passes. SGD is efficient so we do this in batches and we should divide the total number of passes (feedforward + backward) by the batchsize (100). This means the training computational time is 40,000/100 = 400 passes. This is on the same level as what is required during sleep (400-500 passes till convergence). The computational performance of sleep phase can be further improved (e.g., by incorporating the idea of mini-batches, etc.), just as how gradient descent has been heavily optimized.

Lastly, in terms of memory constraints, there are scenarios where SRA reduces storage requirements (as shown with iCaRL) and scenarios where SRA requires more memory (as compared to regularization methods). However, regularization methods may be more computationally intensive, as they require computation of complex matrices, e.g. Fisher information matrix in EWC, or orthogonal projectors in OWM.

4.7.3 Effect of input type during sleep

One possible reason for digit-specific reactivation may be that the average image based input is applied during sleep. Thus, we performed the same analysis presented in the main paper with random inputs (drawn from a uniform distribution) presented during sleep. In Figure 4.7, we show the firing rates of digit specific neurons when presented with such noisy random inputs. We

still observe that digit-specific neurons are more likely to fire during sleep. This suggests that the weight-matrices can support preferential activation of task-specific neurons during sleep.

To further test this hypothesis, we tested SRA for incremental MNIST task with different types of inputs presented during sleep - random input, average input, specific inputs, each presented in 2 conditions (mask/no mask). For random inputs, we build a random uniform vector with the same size as the input space to determine input firing rates of neurons during sleep. For average input, the same input as described in the main paper is used, i.e., the average of all inputs seen so far determines the input firing rates. For specific inputs, we present Poisson distributed spike trains based on specific images (e.g., an image of a 3) during sleep. In the mask condition, during each iteration of sleep we randomly select a 10x10 portion of the input to present during sleep. In the no mask condition, the entire 28x28 vector is used to compute firing rates.

Figure 4.10 shows the classification accuracy for each type of input. For specific inputs classification accuracy reaches ~60% in both the mask and no mask condition. For average inputs, we see a ~10% degradation when the entire image is presented (compared to specific inputs). However, with the mask condition, the degradation between moving from specific to average inputs is much smaller. Finally, looking at random inputs, we observe that when the entire random image is presented, the network is still able to mitigate catastrophic forgetting (task performance is ~30%). However, performance is much worse with random inputs, suggesting that task-specific information must be present for the network to optimally recover older tasks.

4.7.4 Effect of task training length

Our results suggest that the information about old tasks is not lost in the network after new task training, even if from a classification standpoint the old tasks are not classified correctly. One can expect, however, that as a new task training increases, the network may eventually lose all the

old task information. In order to verify this, we measured the classification accuracy before and after sleep in the simplified two task incremental setting for MNIST, Fashion MNIST, CIFAR10, and the crossmodal MNIST task. For MNIST, Fashion MNIST, and CIFAR10, task 1 is the first two classes in the dataset and task 2 is the next two classes. For crossmodal MNIST, task 1 is the entire MNIST task, and task 2 is the entire Fashion MNIST task. We varied the length of training of task 2 in order to test the hypothesis that longer task 2 training would result in an inability for task 1 to be recovered after sleep since there is less information in the network pertaining to task 1.

Figure 4.11 shows classification accuracy on task 1 and task 2 before and after sleep, for all 4 datasets as a function of the length of task 2 training (in epochs). In all cases, T1 is mostly forgotten after T2 training (red dashed line), except in the case of Cross Modal MNIST, where T1 accuracy is around ~20%. With light task 2 training, task 1 classification accuracy is recovered significantly. However, as we increase the length of task 2 training, task 1 recovery decreases. Nevertheless, it never becomes completely forgotten after sleep (solid red line is always above dashed red). Note that the reverse is true for T2. For light T2 training, T2 becomes more degraded after sleep. With longer T2 training, T2 performance remains unchanged after sleep (dashed gray line = solid gray line).

These results also suggest that possibly effective training strategy would be to interleave multiple episodes of new task training and sleep. Indeed, this would well match biology, when new procedural task training in human brain happens slowly and involves multiple training sessions and multiple episodes of sleep between them.

4.7.5 Effect of SRA on single task performance

From neuroscience standpoint the best studied effect of sleep on memory is an improvement of a single task performance after sleep. E.g., in experiments with new skill learning, comparing task performance following episode of training + sleep vs. training with no sleep reveals memory improvement when sleep is included. In fact, recent studies revealed an inverse association between learning performance and gains in procedural skill, i.e., good learners exhibited smaller performance gains after sleep than poor learners (Rångtell et al. 2017).

To test if SRA can provide similar performance benefits for a single task learning and also show similar dependence on pre-sleep performance, we used the entire MNIST dataset and trained it in 4 conditions, that were different by the length of training. SRA was implemented after training in all 4 cases and we compared performance before vs. after sleep.

Figure 4.12 shows the classifcation accuracy on the entire MNIST dataset before/after sleep in 4 training conditions. When training is short, i.e., pre-sleep performance is low, we found a very significant 20 percentage point increase in classification accuracy after SRA. When training is long, i.e., pre-sleep performance is high, we see a ~5 percentage point improvement. Finally, for very long training, we observed small reduction in performance after SRA. The last may be related to the reported SRA role in increasing generalization (Tadros et al. 2019). Indeed, requirements for robustness/generalization and classification accuracy are conflicting, so increase in generalization may explain observed reduction of accuracy after SRA even on recent tasks. These results suggest that replay during SRA can indeed capture some important elements of replay in the biological brain.

4.7.7 Implementation of other methods

4.7.7.1 Elastic Weight Consolidation

Elastic weight consolidation (Kirkpatrick et al. 2017) seeks to reduce the plasticity of weights that are deemed important for previously learned tasks, while utilizing less important weights to learn and represent the new task. Any updates to weights deemed important for previously learned tasks are penalized in the loss function, thus making it harder for any new training to impact the performance of old tasks. More formally, this is written as a regularization term added onto the loss function

$$L(\theta) = L_B(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta^*_{A,i})$$

Here $L_B(\theta)$ represents the normal loss function evaluated on the new training set B. The regularization term computes the product of the Fisher information matrix of parameter i with the difference between the proposed weight update $\theta_i$and the original parameter when trained on an earlier dataset $\theta^*_{A,i}$. In this way, weight updates to parameters deemed important for task A are penalized. The lambda parameter determines how plastic the network should be. A high value for lambda suggests that weight updates should only occur when they are vital to achieving a high performance on the new task B. For our purposes, we performed a parameter sweep over lambda to find the best value of lambda for each specific task.

In this work, we implemented elastic weight consolidation (EWC) as a benchmark to compare against the sleep replay algorithm. In the original work (Kirkpatrick et al. 2017), EWC was tested on the MNIST permutation task as well as a series of reinforcement learning games. In other works, it has been noted that EWC does not achieve optimal results on incremental learning tasks (when the classes in the dataset are learned incrementally) (van de Ven, Siegelmann, and Tolias 2020a). In this work, we corroborate these results. Mainly, EWC fails to perform continual

learning on the incremental learning tasks. However, on the Multi-modal MNIST task where the network must learn both the Fashion MNIST and MNIST datasets, EWC outperforms SRA. These results suggests that different methods can excel in certain areas of continual learning but fail in others and that combinations of different methods may provide an optimal solution that can be explored in future studies.

4.7.7.2 Synaptic Intelligence

The Synaptic Intelligence approach (SI) (Zenke, Poole, and Ganguli 2017) is similar to EWC with a slightly different regularization term. To penalize updates to important parameters, the SI method uses the following loss function:

$$L'_\mu = L_\mu + c \sum_i \Omega_i^\mu (\theta'_i - \theta_i)\text{^}2$$

This regularization term is similar to EWC, where Ω_i^μ keeps track of how important an individual parameter θ_i is to previously learned tasks. The parameter c is similar to the lambda parameter in EWC. We performed a parameter search over c to find the best value for each of the tasks tested in the paper. Given the similarities of the two approaches, we observed similar performances on the tasks tested in the paper between EWC and SI. Mostly, SI fails to perform class incremental learning but performs well on the Multi-modal MNIST task.

4.7.7.3 Orthogonal Weight Modification

The orthogonal weight modification (OWM) approach (Zeng et al. 2019) takes a different approach by trying to force weight updates in a direction that is orthogonal to the subspace spanned by all previously learned inputs. First, a projector P is constructed to find the direction orthogonal to the input space: $P = I - A(A^T A + \alpha I)^{-1}$, where A consists of all previously trained inputs as columns. Weight modifications are then done by $\nabla W = \kappa P \nabla W^{BP}$. We used α values of 0.9*(0.001^l), 0.1^l, and 0.6 in each of the 3 hidden layers. Here, l represents the progress made

through the training of the current task. For CIFAR-10, we used 0.9*(0.0001^l), 0.1^l, and 0.006 as the α parameters. The authors in the original paper use an iterative method to construct P so not all inputs must remain stored. In the tasks tested in the paper, OWM performs significantly better than the other two regularization methods: EWC and SI.

Note that in the manuscript we discuss two different versions of OWM - one with shorter training and another one with longer training. Table 1 in the manuscript shows OWM performance when trained with 2-4 epochs/task (the same as for all other methods) on MNIST, Fashion MNIST and CIFAR10. Table 2 shows OWM performance when trained with 10 epochs/task on all datasets. We discovered that OWM takes longer to converge than most of the other methods tested in this manuscript and so we present data over a range of training duration (see Figure 4.15). The results with longer training are in line with other implementations of OWM except for in the case of CIFAR-10 (Zeng 2019). We hypothesize that this disparity occurs because our study only applies OWM to the fully connected layers, not in an end-to-end fashion, as done in Zeng 2019. Since our extracted features for CIFAR10 images were based on Tiny Imagenet, they are likely not maximally informative. If we use extracted features from a network trained on CIFAR10 instead of Tiny Imagenet, we could improve OWM performance on CIFAR10 (when trained sequentially) from 34% to 42%. This suggests that OWM may work better in scenarios where the feature extractor can also be fine-tuned on the dataset.

4.7.8 Discussion of regularization approaches

Table 1 in the main paper illustrates that the first two regularization (SI and EWC) approaches fail to prevent catastrophic forgetting in the incremental tasks. However, on the Multi-modal MNIST task, these regularization approaches come close to reaching the ideal accuracy. Here, we briefly discuss why these methods may succeed in some domains and fail in others. In

the paper outlining EWC, the authors tested the algorithm on the MNIST permutation task and reinforcement learning tasks (Kirkpatrick et al. 2017). For SI, the algorithm was tested on the MNIST permutation task as well as the split MNIST and CIFAR tasks (Zenke, Poole, and Ganguli 2017).

The MNIST permutation task defines a single task as one representation of the input space. During subsequent tasks, this representation is permuted (so that pixel i of all images now appears in a new location in the image). During incremental learning, the network must be able to distinguish all 10 digits in previously learned permutations, while also learning the newest permutation. Since a lot of the information in MNIST images is present in the center of the image, we hypothesize that when a new permutation of the images is created, the pertinent information is likely moved to distinct regions of the image. These regions have a relatively small amount of overlap with the old permutations, so when EWC or SI are applied, important weights connecting from the center of the image may be preserved, while the weights that connect from the new important regions of the image are updated.

For split MNIST/CIFAR tasks, the classes in the dataset are divided into subsets (i.e. pairs of digits or images), and the network learns to distinguish each image in the pair. During task 1, the network may learn to distinguish 0 and 1. Then, during task 2, the network may learn to distinguish 2 and 3, utilizing the same two output neurons as were used for task 1. In this scenario, regularization approaches may work well by allocating distinct pathways through the network for each binary classification task learned. Thus, when task 2 is learned, old pathway through the network is preserved for task 1, and a new pathway is created to represent task 2. In incremental learning settings, activity to certain output neurons is cut off during new task training, as is the case with the incremental tasks tested in the paper. Thus, when training on a new task (e.g., digits

2 and 3), output neurons representing old memories (e.g., digits 0 and 1) should not receive any activity to achieve a suitable loss on the new task. We hypothesize that the regularization term added by EWC and SI is insufficient to maintain activity to these output neurons representing older classes.

Nevertheless, we found that EWC and SI work well for the multi-modal MNIST task (where the network must learn MNIST and Fashion MNIST incrementally). In this task, during both phases of learning, all output neurons remain active. Additionally, the MNIST and Fashion MNIST datasets may maintain information in distinct parts of the image. Therefore, regularization approaches may consolidate the two tasks into the network, solely by preserving weights that are significantly important for the old task and using the relatively less important weights to represent the new task.

4.7.9 Incremental Classifier and Representation Learning (iCaRL)

Here, we describe our implementation of the iCaRL method and differences with the main paper (Rebuffi et al. 2017). The main idea of the iCaRL algorithm is to utilize a fixed memory capacity (capacity K, where K represents number of examples stored from previous classes) in an efficient way to prevent catastrophic forgetting. Specifically, it selects which K examples to store from previous classes (m = K/t images per class, where t is the number of classes seen so far) by adding examples to the memory bank which cause the average feature vector over all exemplars (m images in the memory bank) to approximate the average feature vector over all training examples (all images of the relevant class). These images are sorted in the exemplar set based on how well they approximate the average feature vector over all training examples. When a new class is learned, the exemplar set for a certain class is reduced (now m = K/(t+1) by removing the least informative (of the average feature vector) examples.

In addition to introducing an efficient exemplar management, the learning rule and classification scheme are modified in the original iCaRL implementation (Rebuffi et al. 2017). Specifically, the learning rule incorporates both soft-target distillation over examples from previous tasks as well as cross-entropy loss from the newest task. The labels for the new images are binary one-hot encoded vectors with the correct classifications. However, the labels for the old images are computed by passing these images through the network and storing the output from the network. In addition, the classification scheme is changed to use the Nearest Means of Exemplar classification strategy, where test inputs are fed through the network and their representation in the last hidden layer (before the classification layer) is used to compute the classification for that test example. Its label is determined by finding the nearest-class-mean using the exemplar set and all class labels which have been learned so far. In our implementation of iCaRL, we used the Nearest Means of Exemplar classifier as well as soft-target distillation.

As with OWM, we found that iCaRL can benefit from longer training times (see Figure 4.16). We, however, showed that even with longer training times, SRA can still improve upon iCaRL performance, especially for lower values of K. Furthermore, SRA can reduce the training time needed to achieve maximal performance of iCaRL (Figure 4.16). We characterize the savings in the training time by the difference in epochs/task when iCaRL + SRA achieves the same accuracy as iCaRL alone after 10 training epochs. These results are reported in the main manuscript.

4.7.10 Implementation of both training and sleep replay within SNN

Sleep replay algorithm presented in this study was implemented for artificial neural networks. In addition, we tested another implementation when both training and sleep were implemented using spiking neural networks (SNNs). SNNs recently received attention from the

neuromorphic hardware community for their ability to perform power-efficient computing (Ghosh-Dastidar and Adeli 2009), particularly during inference phase (P. U. Diehl et al. 2015; Zhang et al. 2019; Davidson and Furber 2021). However, SNNs are still inferior to ANNs on complex data training (Wu et al. 2019). Nevertheless, the question is raised of whether or not the entire wake-sleep pipeline could be performed within a spiking network architecture as well as what implications this may have for neuroscience and computer science.

Table 4.4: Catastrophic forgetting is observed when sequential training is performed in an SNN architecture

SRA is able to produce more balanced classification accuracy on both tasks (T1 = digits 0-1, T2 = digits 2-3 in MNIST dataset). Each value represents the average of 5 trials.

Phase	T1 Performance	T2 Performance
T1 awake training	99.6	0.6
Sleep	84.8	2.9
T2 awake training	0	85.9
Sleep	71.8	51.3

To address this question, we implemented both awake training and sleep replay withing SNN architecture. We performed "awake" sequential training on MNIST data in an SNN via backpropagation (Yanguas-Gil 2020). This method works by approximating the discontinuity in a neuron's non-linear spiking function so as to make it differentiable when performing backpropagation. Our goals were (a) to verify that catastrophic forgetting occurs in an SNN trained through backpropagation; (b) to test the effect of SRA in overcoming catastrophic forgetting.

Table 4.4 illustrates the classification accuracy for two task incremental training on the MNIST dataset (T1 = digits 0-1, T2 = digits 2-3). The SNN trained here had the same number of neurons in each layer as the ANNs used in the main paper (2 hidden layers with 1200 neurons each). After training on T1, SNN was able to classify images in T1 with 99% classification accuracy (averaged over 5 trials). After training on T2, catastrophic forgetting occurred, as classification accuracy on T1 dropped to 0%. This illustrates that catastrophic forgetting is not

unique to ANNs but can occur in SNNs due to the interference between the tasks. (Indeed, our recent studies suggest that catastrophic forgetting can occur in a spiking network even with biologically realistic local learning rules (Gonzalez et al. 2019; Golden et al. 2020) when tasks with a high amount of overlap are learned incrementally.) After sleep, performance on T1 was mostly recovered while performance on T2 was partially reduced. Overall, this experiment demonstrates that catastrophic forgetting does occur in SNNs and sleep can reduce the impact of catastrophic interference.

4.7.11 Analysis of sleep replay

To evaluate replay of the old and new tasks during sleep, we compared the neurons that spiked frequently during sleep with the average activation in the ANN for each specific image (Figure 4.13) When an image was presented to the ANN (before SRA is applied), we recorded the activation vectors in each hidden layer $a_{1,i,j}$, $a_{2,i,j}$, for hidden layers 1 and 2, respectively, for each image i of class j. For two tasks, we recorded values for $a_{1,i,0}$ through $a_{1,i,3}$, and $a_{2,i,0}$ through $a_{2,i,3}$ to store activation vectors for each class of digits learned, 0 through 3. Then, we tested for the overlap between neurons that had non-zero activation values in ANN (before SRA) for a specific input and neurons which had non-zero firing rates during subsequent sleep (during SRA). This serves as a proxy to measure the extent to which a specific input is replayed during sleep. As a control group, we performed the same calculations but using ANN activation vectors obtained in response to an image created with uniformly-distributed random noise.

Figure 4.13A shows the average amount of reactivation for each digit in each hidden layer, normalized by the reactivation of the control group. In layer 1, for each digit, we observed reactivation values that were greater than reactivation for a random input. Most notably, digit 1 (from the old task) seems to be reactivated the most, followed by digits 2 and 0. Analysing the

confusion matrices before and after sleep for this specific network (Fig 4.13B), we observed that the classification accuracy for digit 1 was particularly improved after sleep, and when a network predicted an image incorrectly, it often misclassified it as digit 1. This suggests that the digit-specific reactivation correlates well with the network classification performance and further steps to improve reactivation during sleep may lead to further SRA performance improvements. In Layer 2 (Fig 4.14A, right), we observed that all digits were reactivated greater than expected for a random input image. However, the ratio of reactivation for each digit was about equal. Overall, these results suggest that previously learned tasks are replayed during sleep phase and amount of replay positively correlates with classification performance on the old digits that otherwise would have been forgotten.

In addition to reactivation, we also measured the effect of sleep replay on the representation of distinct digits in the network. Work by (McClelland, McNaughton, and O'Reilly 1995) shows that during training, representations of distinct image classes can become more dissimilar and sparser. We perform a similar analysis here by plotting the activation of a group of 40 randomly selected neurons before and after sleep, when presented with each of the 10 MNIST digits (Fig 4.15) Before sleep, the representation of all digits is very similar (with the exception of digits 8 and 9, which were more recently learned). After sleep, every digit has a unique representational code, along with much sparser activity.

4.8 Figures

Algorithm 1 Sleep:

procedure ANNTOSLEEPANN(nn)
  Change ReLU activation in ANN to Heaviside function in SleepANN and determine layer-wide specific threshold
  Apply weight normalization and return scale, threshold for each layer **return** $SleepANN, scales, threshold$
procedure SLEEPANNTOANN(nn)
  Directly map the new, unscaled weights from Heaviside-network (SleepANN) to ReLU network (ANN) **return** ann
procedure SLEEP($nn, I, scales, thresholds$) ▷ I is input
  Initialize v (voltage) $= 0$ vectors for all neurons
  for $t \leftarrow 1$ to Ts **do** ▷ Ts - duration of sleep
    $\mathbf{S}(1) \leftarrow$ Convert input I to Poisson-distributed spiking activity
    $\mathbf{S}$ = ForwardPass($\mathbf{S}, v, \mathbf{W}, scales, thresholds$)
    $\mathbf{W}$ = BackwardPass($\mathbf{S}, \mathbf{W}$)
procedure FORWARDPASS($S, v, W, scales, threshold$)
  for $l \leftarrow 2$ to n **do** ▷ n - number of layers
    $v(l) \leftarrow v(l) + (scales(l-1)\mathbf{W}(l, l-1)\mathbf{S}(l-1))$ ▷ W(l,l-1) - weights
    $\mathbf{S}(l) \leftarrow v(l) > threshold(l)$ ▷ Propagate spikes
    $v(l)(v(l) > threshold(l)) = 0$ ▷ Reset spiking neurons' voltages
  return S
procedure BACKWARD PASS(S, W)
  for $l \leftarrow 2$ to n **do** ▷ n - number of layers
    $$\mathbf{W}(l, l-1) \leftarrow \begin{cases} \mathbf{W}(l, l-1) + inc & \text{if } \mathbf{S}(l,t) = 1 \,\&\, \mathbf{S}(l-1,t) = 1 \\ \mathbf{W}(l, l-1) - dec & \text{if } \mathbf{S}(l,t) = 1 \,\&\, \mathbf{S}(l-1,t) = 0 \end{cases}$$ ▷ STDP
  return W
procedure MAIN
  Initialize neural network (ANN) with ReLU neurons and bias $= 0$.
  for task $t = 1 : T$ **do**
    Train ANN using backpropagation on task t.
    $SleepANN, scales, thresholds$ = ANNtoSleepANN(ANN)
    $SleepANN$ = Sleep($SleepANN$, Training data X, $scales$, $thresholds$)
    ANN = SleepANNtoANN($SleepANN$)

Figure 4.1: Sleep Replay Algorithm pseudocode for catastrophic forgetting

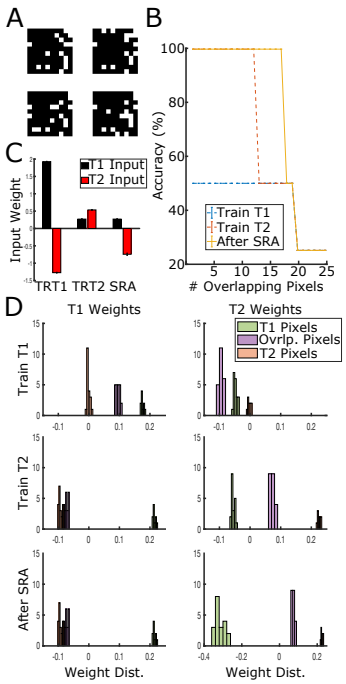

Figure 4.2: SRA reduces catastrophic forgetting for sequential task training

(A) Example of four binary images with 15 pixel overlap divided into task 1 (T1, images #0,1) and task 2 (T2, images #2,3). T2 was trained after T1.

(B) Classification accuracy as a function of number of overlapping pixels (interference) after training T1 (blue), T2 (red), and SRA (yellow). The network was always tested for all 4 images. Note, that performance is significantly higher after SRA than before in the range of pixel overlap 10-20.

(C) Input to Output Neuron 0 when presented with Image 0 (black) and max input to Output Neurons 2 and 3 when presented with image 0 (red) after subsequent stages of training and sleep.

(D) Distribution of weights connecting to T1 (left) or T2 (right) output neurons. Color is based on category of connections: Unique - connections from neurons representing pixels unique for T1 or T2; Overlapping - connections from neurons representing overlapping pixels between T1 and T2. Rows show subsequent stages of training and sleep.

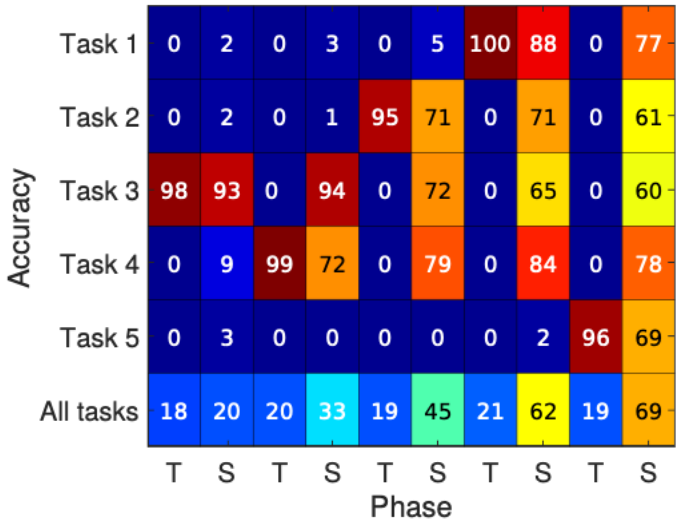

Figure 4.3: Interleaving new training with sleep results in recovery of old task performance on MNIST dataset

Sleep phase (S) was implemented after each new task training (T). Task 1 - 0/1, Task 2 - 2/3, Task 3 - 4/5, Task 4 - 6/7, Task 5 - 8/9. Each column shows performance for all tasks after either new task training or sleep as labeled below.

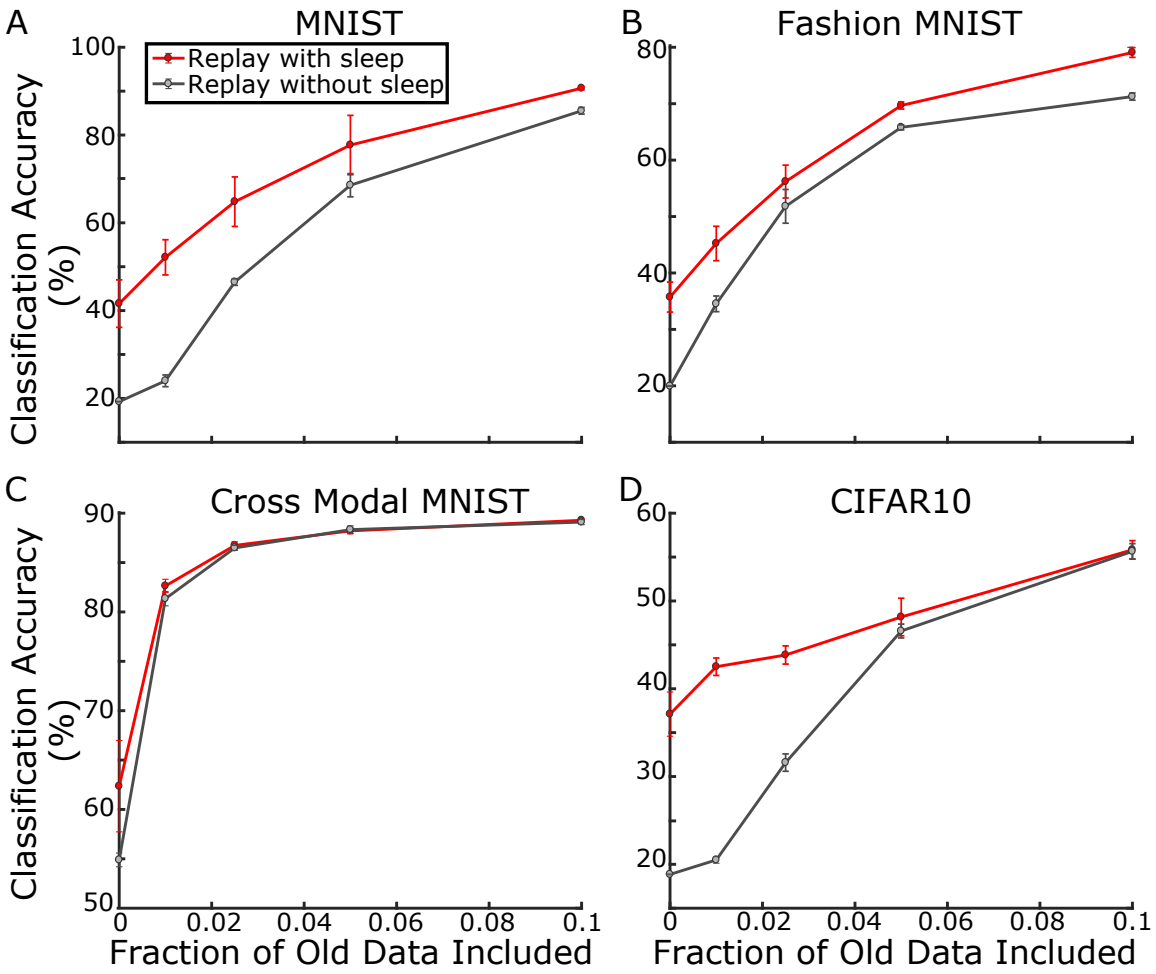

Figure 4.4: SRA is complementary to replay methods which store/generate old examples and combine new data with these examples during training

The classification accuracy as a function of percent of old data included is shown for A) MNIST, B) Fashion MNIST, C) Cross Modal MNIST, and D) CIFAR10. Error bars represent standard deviations across 5 training sessions with different task orders.

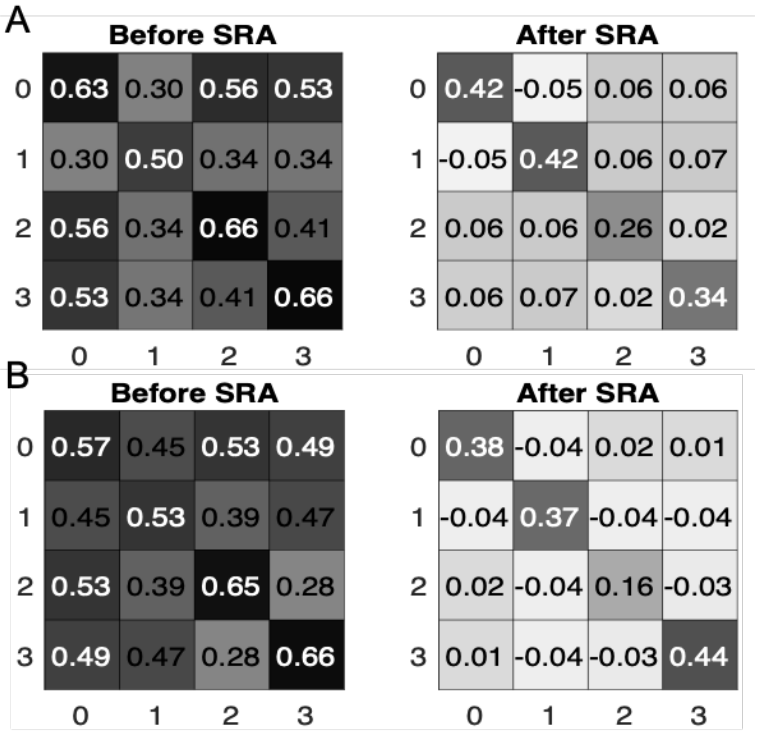

Figure 4.5: SRA reduces correlations between image classes while maintaining strong correlations within classes

Correlation matrices of activations in hidden layer 1 (A) and layer 2 (B). Labels (0-3) indicate image class: Task 1 - 0/1, Task 2 - 2/3. Note that before SRA (left) correlations between classes, e.g., 1 and 2 images, are almost as high as correlations within classes, e.g., images of 1.

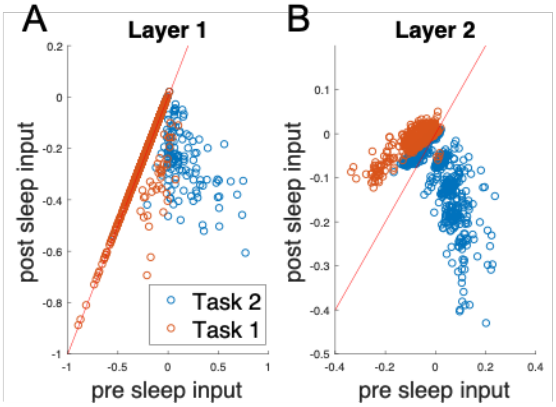

Figure 4.6: SRA changes input to the hidden layer neurons to favour old tasks

In layer 1 (left), input to the neurons representing old task is reduced less than input to the neurons representing most recently learned task. In layer 2 (right), SRA increases input to the neurons representing old task while decreasing input to the neurons representing most recently learned task.

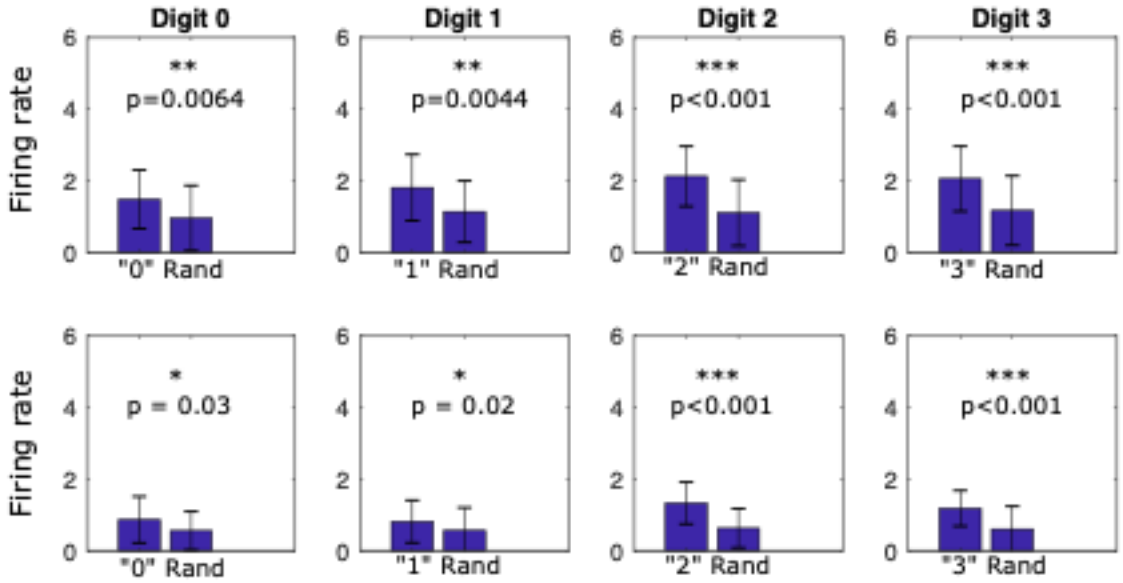

Figure 4.7: Spiking activity of digit-specific neurons during sleep is greater than activity of a random subset of neurons in both first (A) and second (B) hidden layers.

Neurons are activated by random (not average) inputs to the input layer. Error bars represent standard deviation. p-values computed from two-sided t-test comparing firing rates of digit-specific neurons vs. random subset of neurons with Bonferroni Correction.

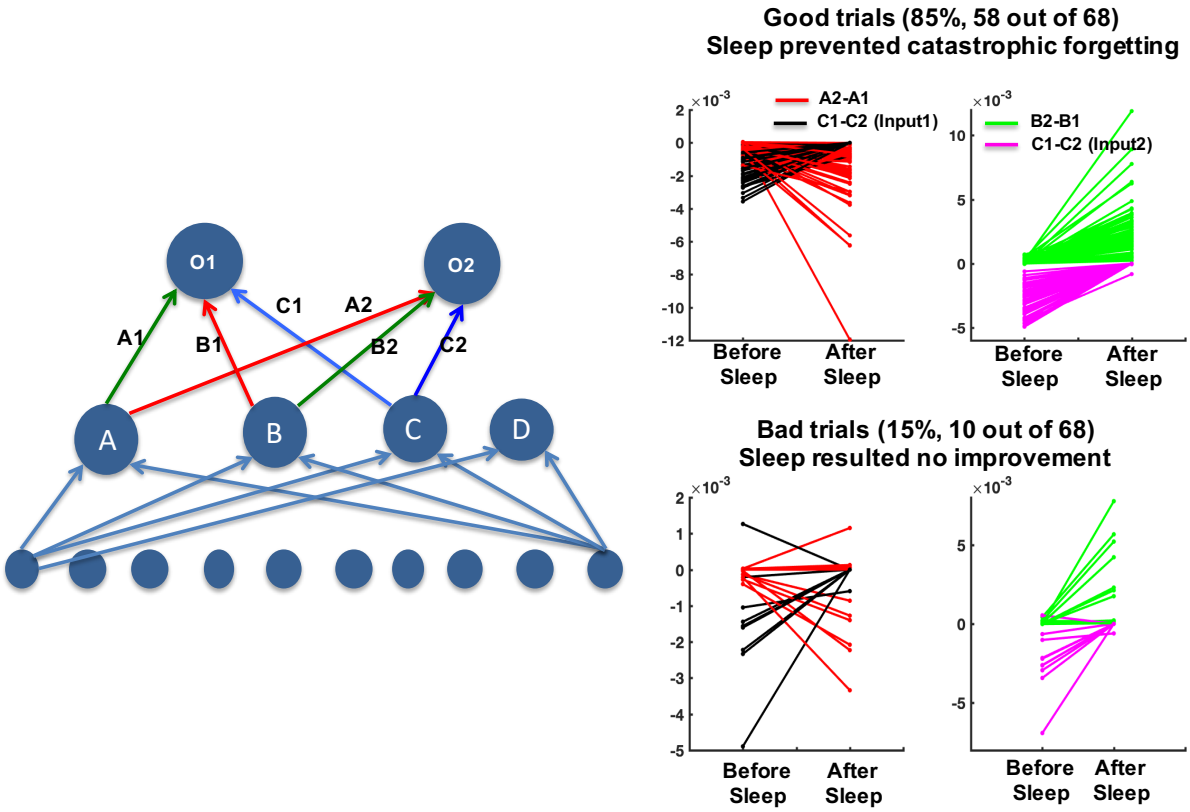

Figure 4.8: Example of binary vector analysis

In the left graph, we show the structure of the network. A - fires only for input 1. B - fires only for input 2. C - fires for both inputs. D - fires for neither input 1 nor input 2. Green arrows represent desirable connections and red arrows indicate incorrect connections. Blue arrows are mixed depending on the input. The equations on the graphs on the right compare the difference between green and red arrows to the difference.

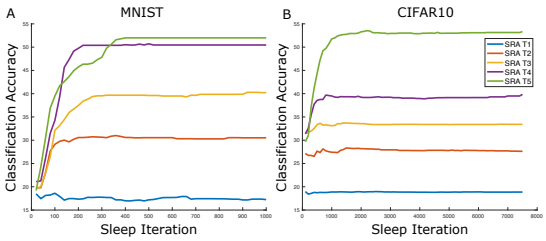

Figure 4.9: Accuracy as a function of sleep iteration for MNIST (left) and CIFAR10 (right) datasets

Each line represents the application of SRA after a specific training phase (e.g., after training T1, T2, etc).

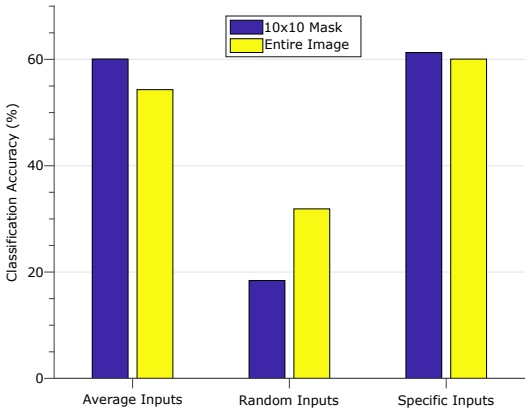

Figure 4.10: Classification accuracy on class-incremental MNIST task with different types of inputs (average, random, specific) with 2 conditions (mask/no mask)

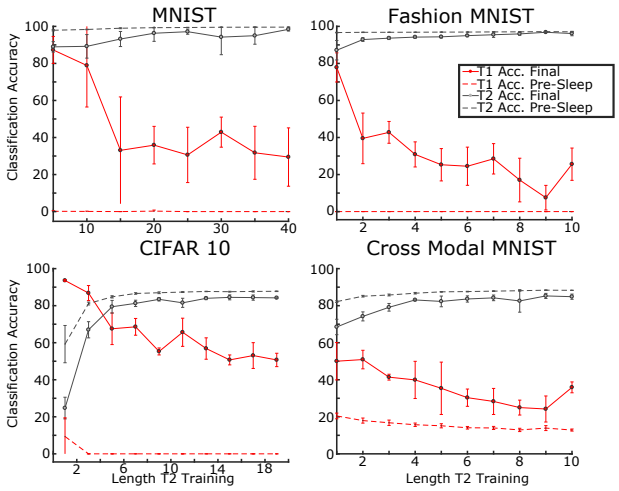

Figure 4.11: Classification accuracy as a function of length of task 2 training (in epochs) for 2 task MNIST, Fashion MNIST, and CIFAR 10 as well as the Cross Modal MNIST task

Solid red - T1 accuracy after sleep, dashed red - T1 accuracy before sleep (after T2 training), dashed gray - T2 accuracy before sleep (after T2 training), and solid gray - T2 accuracy after sleep.

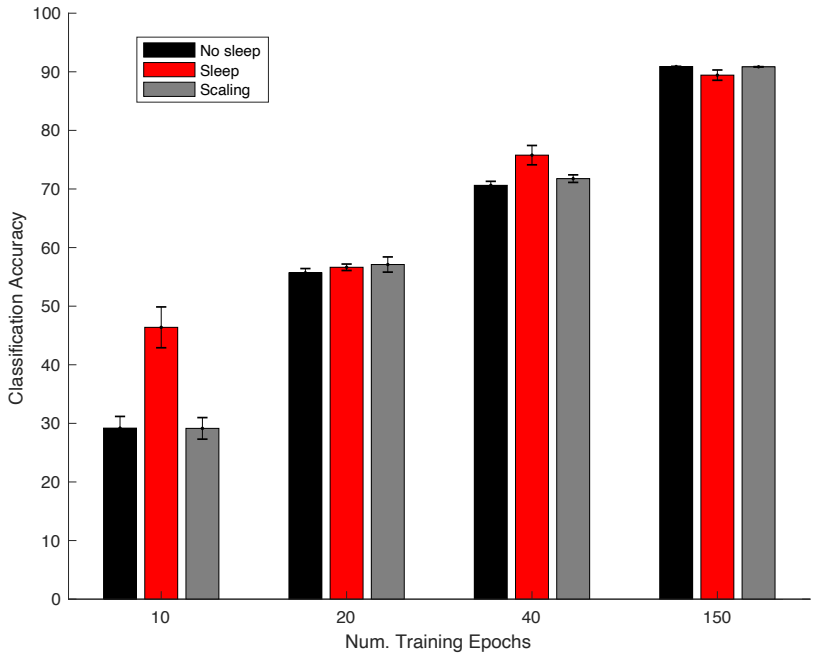

Figure 4.12: Undertrained MNIST classification accuracy

Classification accuracy on the MNIST dataset before (black) and after (red) sleep when trained for 10, 20, or 40, 150 epochs. Scaling of top 2.5% of weights by 1% shown in grey.

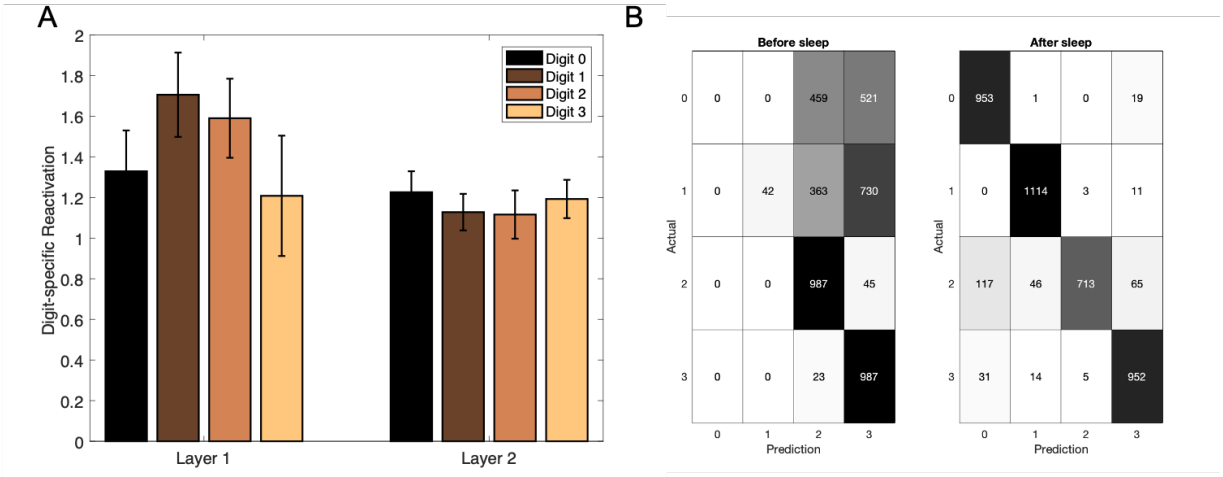

Figure 4.13: MNIST digit-specific reactivation analysis

During sleep, digit-specific activations are replayed more often than would be expected by chance. A) Digit-specific reactivation in layer 1 (left) and layer 2 (right) for each digit. Each value is normalized by a control group so values greater than 1 indicate there is more reactivation than would be expected for a randomly generated image. B) Confusion matrices before (left) and after (right) sleep for the two tasks.

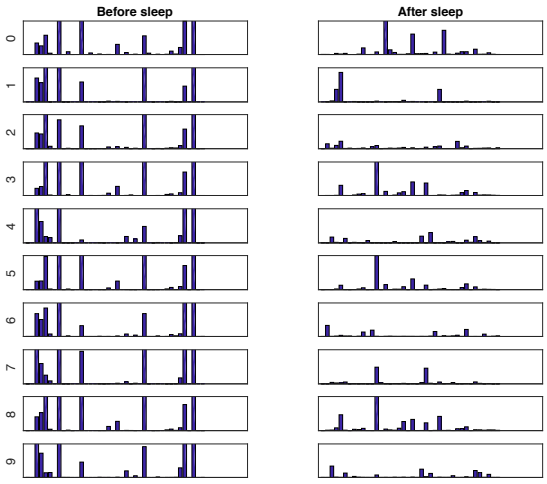

Figure 4.14: SRA results in sparsification of task representation

Average activation of 40 randomly selected neurons before (left) and after sleep (right) when presented with digits from each class. Overall, less neurons respond to each digit suggesting a sparser representation

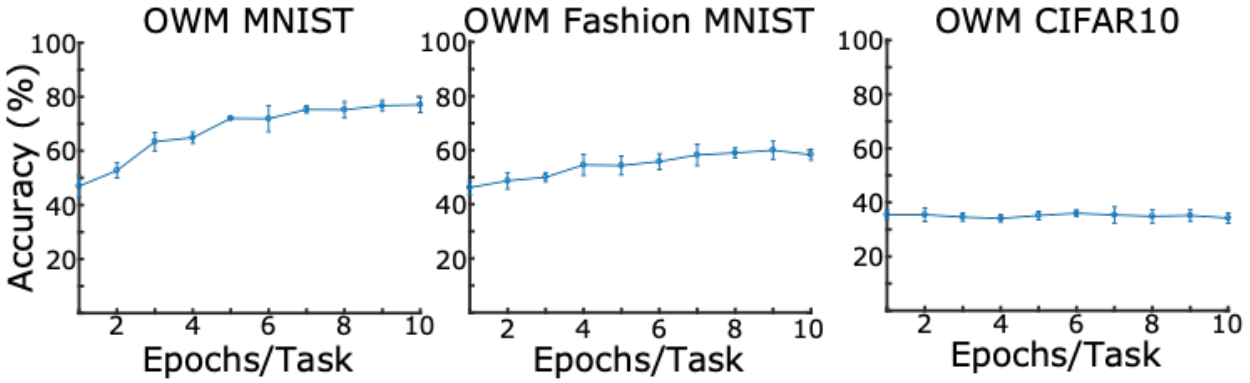

Figure 4.15: Analysis of OWM performance as a function of number of epochs per task on MNIST (left), Fashion MNIST (middle) and CIFAR10 (right) datasets

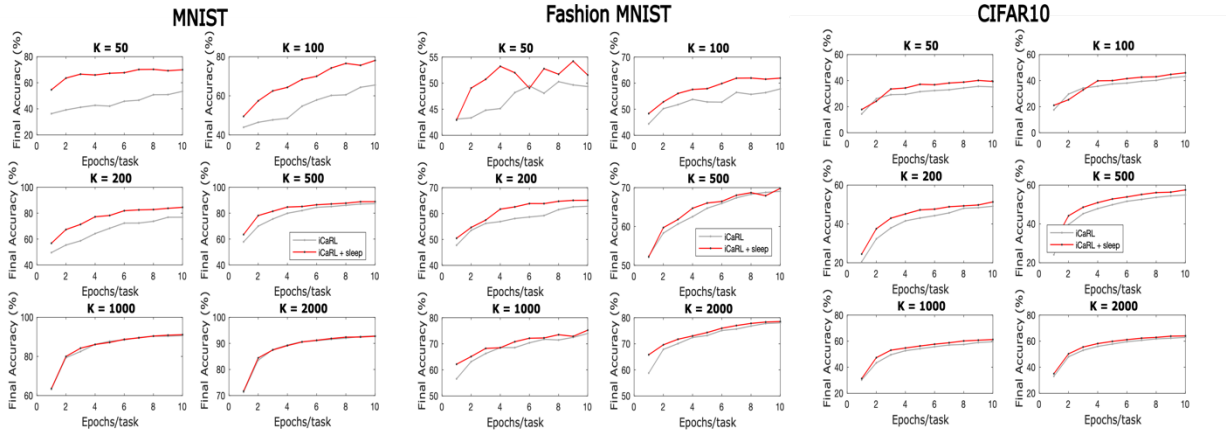

Figure 4.16: Analysis of iCaRL performance with (red) and without (gray) SRA as a function of number of epochs per task on MNIST (left), Fashion MNIST (middle) and CIFAR10 (right) dataset

4.9 References

Bazhenov, Maxim, Igor Timofeev, Mircea Steriade, and Terrence J Sejnowski. 2002. "Model of Thalamocortical Slow-Wave Sleep Oscillations and Transitions to Activated States." *Journal of Neuroscience* 22 (19): 8691–8704.

Bouchacourt, Flora, Stefano Palminteri, Etienne Koechlin, and Srdjan Ostojic. 2020. "Temporal Chunking as a Mechanism for Unsupervised Learning of Task-Sets." *Elife* 9: e50469.

Davidson, Simon, and Steve B Furber. 2021. "Comparison of Artificial and Spiking Neural Networks on Digital Hardware." *Frontiers in Neuroscience* 15.

Dayan, Peter, and Geoffrey E Hinton. 1996. "Varieties of Helmholtz Machine." *Neural Networks* 9 (8): 1385–1403.

Diehl, Peter U, and Matthew Cook. 2015. "Unsupervised Learning of Digit Recognition Using Spike-Timing-Dependent Plasticity." *Frontiers in Computational Neuroscience* 9: 99.
Diehl, Peter U, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. 2015. "Fast-Classifying, High-Accuracy Spiking Deep Networks through Weight and Threshold Balancing." In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8. ieee.

Euston, David R, Masami Tatsuno, and Bruce L McNaughton. 2007. "Fast-Forward Playback of Recent Memory Sequences in Prefrontal Cortex during Sleep." *Science* 318 (5853): 1147–50.

French, Robert M. 1999. "Catastrophic Forgetting in Connectionist Networks." *Trends in Cognitive Sciences* 3 (4): 128–35.

Gerstner, Wulfram, Werner M Kistler, Richard Naud, and Liam Paninski. 2014. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press.

Ghosh-Dastidar, Samanwoy, and Hojjat Adeli. 2009. "Spiking Neural Networks." *International Journal of Neural Systems* 19 (04): 295–308.

Gil, Ziv, Barry W Connors, and Yael Amitai. 1997. "Differential Regulation of Neocortical Synapses by Neuromodulators and Activity." *Neuron* 19 (3): 679–86.

Giri, Bapun, Hiroyuki Miyawaki, Kenji Mizuseki, Sen Cheng, and Kamran Diba. 2019. "Hippocampal Reactivation Extends for Several Hours Following Novel Experience." *Journal of Neuroscience* 39 (5): 866–75.

Golden, Ryan, Jean Erik Delanois, Pavel Sanda, and Maxim Bazhenov. 2020. "Sleep Prevents Catastrophic Forgetting in Spiking Neural Networks by Forming Joint Synaptic Weight Representations." *BioRxiv*, 688622.

González, Oscar C, Yury Sokolov, Giri Krishnan, and Maxim Bazhenov. 2019. “Can Sleep Protect Memories from Catastrophic Forgetting?” *BioRxiv*, 569038.

Goodfellow, Ian J, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. “An Empirical Investigation of Catastrophic Forgetting in Gradient-Based Neural Networks.” *ArXiv Preprint ArXiv:1312.6211*.

Hayes, Tyler L, Nathan D Cahill, and Christopher Kanan. 2019. “Memory Efficient Experience Replay for Streaming Learning.” In *2019 International Conference on Robotics and Automation (ICRA)*, 9769–76. IEEE.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. “Deep Residual Learning for Image Recognition.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–78.

Hennevin, Elizabeth, Bernard Hars, Catherine Maho, and Vincent Bloch. 1995. “Processing of Learned Information in Paradoxical Sleep: Relevance for Memory.” *Behavioural Brain Research* 69 (1–2): 125–35.

Hill, Sean, and Giulio Tononi. 2005. “Modeling Sleep and Wakefulness in the Thalamocortical System.” *Journal of Neurophysiology* 93 (3): 1671–98.

Hinton, Geoffrey E, Peter Dayan, Brendan J Frey, and Radford M Neal. 1995. “The" Wake-Sleep" Algorithm for Unsupervised Neural Networks.” *Science* 268 (5214): 1158–61.

Javed, Khurram, and Faisal Shafait. 2018. “Revisiting Distillation and Incremental Classifier Learning.” In *Asian Conference on Computer Vision*, 3–17. Springer.

Ji, Daoyun, and Matthew A Wilson. 2007. “Coordinated Memory Replay in the Visual Cortex and Hippocampus during Sleep.” *Nature Neuroscience* 10 (1): 100–107.

Kemker, Ronald, and Christopher Kanan. 2017. “Fearnet: Brain-Inspired Model for Incremental Learning.” *ArXiv Preprint ArXiv:1711.10563*.

Kemker, Ronald, Marc McClure, Angelina Abitino, Tyler L Hayes, and Christopher Kanan. 2018. “Measuring Catastrophic Forgetting in Neural Networks.” In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Kirkpatrick, James, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, et al. 2017. “Overcoming Catastrophic Forgetting in Neural Networks.” *Proceedings of the National Academy of Sciences* 114 (13): 3521–26.

Koch, Christof, and Shimon Ullman. 1987. “Shifts in Selective Visual Attention: Towards the Underlying Neural Circuitry.” In *Matters of Intelligence*, 115–41. Springer.

Kriegeskorte, Nikolaus. 2015. “Deep Neural Networks: A New Framework for Modeling Biological Vision and Brain Information Processing.” *Annual Review of Vision Science* 1: 417–46.

Krishnan, Giri P, Sylvain Chauvette, Isaac Shamie, Sara Soltani, Igor Timofeev, Sydney S Cash, Eric Halgren, and Maxim Bazhenov. 2016. “Cellular and Neurochemical Basis of Sleep Stages in the Thalamocortical Network.” *Elife* 5: e18607.

Krizhevsky, Alex, and Geoffrey Hinton. 2009. “Learning Multiple Layers of Features from Tiny Images.”

Le, Ya, and Xuan Yang. 2015. “Tiny Imagenet Visual Recognition Challenge.” *CS 231N* 7 (7): 3.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. 2015. “Deep Learning.” *Nature* 521 (7553): 436–44.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. “Gradient-Based Learning Applied to Document Recognition.” *Proceedings of the IEEE* 86 (11): 2278–2324.

Lee, Dale K, Laurent Itti, Christof Koch, and Jochen Braun. 1999. “Attention Activates Winner-Take-All Competition among Visual Filters.” *Nature Neuroscience* 2 (4): 375–81.

Lewis, Penelope A, and Simon J Durrant. 2011. “Overlapping Memory Replay during Sleep Builds Cognitive Schemata.” *Trends in Cognitive Sciences* 15 (8): 343–51.

Lewis, Penelope A, Günther Knoblich, and Gina Poe. 2018. “How Memory Replay in Sleep Boosts Creative Problem-Solving.” *Trends in Cognitive Sciences* 22 (6): 491–503.

Li, Zhizhong, and Derek Hoiem. 2017. “Learning without Forgetting.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (12): 2935–47.

Louie, Kenway, and Matthew A Wilson. 2001. “Temporally Structured Replay of Awake Hippocampal Ensemble Activity during Rapid Eye Movement Sleep.” *Neuron* 29 (1): 145–56.

McClelland, James L, Bruce L McNaughton, and Randall C O’Reilly. 1995. “Why There Are Complementary Learning Systems in the Hippocampus and Neocortex: Insights from the Successes and Failures of Connectionist Models of Learning and Memory.” *Psychological Review* 102 (3): 419.

McCloskey, Michael, and Neal J Cohen. 1989. “Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem.” In *Psychology of Learning and Motivation*, 24:109–65. Elsevier.

McCormick, David A. 1992. “Neurotransmitter Actions in the Thalamus and Cerebral Cortex and Their Role in Neuromodulation of Thalamocortical Activity.” *Progress in Neurobiology* 39 (4): 337–88.

McDevitt, Elizabeth A, Katherine A Duggan, and Sara C Mednick. 2015. “REM Sleep Rescues Learning from Interference.” *Neurobiology of Learning and Memory* 122: 51–62.

Mednick, Sara C, Denise J Cai, Tristan Shuman, Stephan Anagnostaras, and John T Wixted. 2011. “An Opportunistic Theory of Cellular and Systems Consolidation.” *Trends in Neurosciences* 34 (10): 504–14.

Mermillod, Martial, Aurélia Bugaiska, and Patrick Bonin. 2013. “The Stability-Plasticity Dilemma: Investigating the Continuum from Catastrophic Forgetting to Age-Limited Learning Effects.” *Frontiers in Psychology* 4: 504.

O’Neill, Joseph, Timothy J Senior, Kevin Allen, John R Huxter, and Jozsef Csicsvari. 2008. “Reactivation of Experience-Dependent Cell Assembly Patterns in the Hippocampus.” *Nature Neuroscience* 11 (2): 209–15.

Oudiette, Delphine, James W Antony, Jessica D Creery, and Ken A Paller. 2013. “The Role of Memory Reactivation during Wakefulness and Sleep in Determining Which Memories Endure.” *Journal of Neuroscience* 33 (15): 6672–78.

Paller, Ken A, and Joel L Voss. 2004. “Memory Reactivation and Consolidation during Sleep.” *Learning & Memory* 11 (6): 664–70.

Ramasesh, Vinay V, Ethan Dyer, and Maithra Raghu. 2020. “Anatomy of Catastrophic Forgetting: Hidden Representations and Task Semantics.” *ArXiv Preprint ArXiv:2007.07400*.

Rångtell, Frida H, Swathy Karamchedu, Peter Andersson, Lieve van Egmond, Tyra Hultgren, Jan-Erik Broman, Jonathan Cedernaes, and Christian Benedict. 2017. “Learning Performance Is Linked to Procedural Memory Consolidation across Both Sleep and Wakefulness.” *Scientific Reports* 7 (1): 1–8.

Rasch, Björn, and Jan Born. 2013. “About Sleep’s Role in Memory.” *Physiological Reviews*.
Rebuffi, Sylvestre-Alvise, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017.

“Icarl: Incremental Classifier and Representation Learning.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001–10.

Robins, Anthony. 1995. “Catastrophic Forgetting, Rehearsal and Pseudorehearsal.” *Connection Science* 7 (2): 123–46.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, et al. 2015. “Imagenet Large Scale Visual Recognition Challenge.” *International Journal of Computer Vision* 115 (3): 211–52.

Saxe, Andrew, Stephanie Nelli, and Christopher Summerfield. 2021. “If Deep Learning Is the Answer, What Is the Question?” *Nature Reviews Neuroscience* 22 (1): 55–67.

Schwindel, C Daniela, and Bruce L McNaughton. 2011. “Hippocampal–Cortical Interactions and the Dynamics of Memory Trace Reactivation.” In *Progress in Brain Research*, 193:163–77. Elsevier.

Shin, Hanul, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. “Continual Learning with Deep Generative Replay.” In *Advances in Neural Information Processing Systems*, 2990–99.
Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, et al. 2018. “A General Reinforcement Learning Algorithm That Masters Chess, Shogi, and Go through Self-Play.” *Science* 362 (6419): 1140–44.

Sonni, Akshata, and Rebecca MC Spencer. 2015. “Sleep Protects Memories from Interference in Older Adults.” *Neurobiology of Aging* 36 (7): 2272–81.

Steriade, Mircea, David A McCormick, and Terrence J Sejnowski. 1993. “Thalamocortical Oscillations in the Sleeping and Aroused Brain.” *Science* 262 (5134): 679–85.

Stickgold, Robert. 2005. “Sleep-Dependent Memory Consolidation.” *Nature* 437 (7063): 1272–78.

———. 2013. “Parsing the Role of Sleep in Memory Processing.” *Current Opinion in Neurobiology* 23 (5): 847–53.

Stickgold, Robert, LaTanya James, and J Allan Hobson. 2000. “Visual Discrimination Learning Requires Sleep after Training.” *Nature Neuroscience* 3 (12): 1237–38.

Tadros, Timothy, Giri Krishnan, Ramyaa Ramyaa, and Maxim Bazhenov. 2019. “Biologically Inspired Sleep Algorithm for Increased Generalization and Adversarial Robustness in Deep Neural Networks.” In *International Conference on Learning Representations*.

Teyler, TJ, and P DiScenna. 1986. “The Hippocampal Memory Indexing Theory.” *Behav Neurosci* 100: 147–54.

Thiele, Johannes, Peter Diehl, and Matthew Cook. 2017. “A Wake-Sleep Algorithm for Recurrent, Spiking Neural Networks.” *ArXiv Preprint ArXiv:1703.06290*.

Timofeev, Igor, and Sylvain Chauvette. 2017. “Sleep Slow Oscillation and Plasticity.” *Current Opinion in Neurobiology* 44: 116–26.

Tononi, Giulio, and Chiara Cirelli. 2006. “Sleep Function and Synaptic Homeostasis.” *Sleep Medicine Reviews* 10 (1): 49–62.

———. 2014. “Sleep and the Price of Plasticity: From Synaptic and Cellular Homeostasis to Memory Consolidation and Integration.” *Neuron* 81 (1): 12–34.

Ven, Gido M van de, Hava T Siegelmann, and Andreas S Tolias. 2020. “Brain-Inspired Replay for Continual Learning with Artificial Neural Networks.” *Nature Communications* 11 (1): 1–14.

Ven, Gido M van de, and Andreas S Tolias. 2018. “Generative Replay with Feedback Connections as a General Strategy for Continual Learning.” *ArXiv Preprint ArXiv:1809.10635*.
Walker, Matthew P, and Robert Stickgold. 2004. “Sleep-Dependent Learning and Memory Consolidation.” *Neuron* 44 (1): 121–33.

Wei, Yina, Giri P Krishnan, and Maxim Bazhenov. 2016. “Synaptic Mechanisms of Memory Consolidation during Sleep Slow Oscillations.” *Journal of Neuroscience* 36 (15): 4231–47.

Wei, Yina, Giri P Krishnan, Maxim Komarov, and Maxim Bazhenov. 2018. “Differential Roles of Sleep Spindles and Sleep Slow Oscillations in Memory Consolidation.” *PLoS Computational Biology* 14 (7): e1006322.

Welinder, Peter, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. 2010. “Caltech-UCSD Birds 200.”

Wilson, Matthew A, and Bruce L McNaughton. 1994. “Reactivation of Hippocampal Ensemble Memories during Sleep.” *Science* 265 (5172): 676–79.

Wu, Yujie, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. 2019. “Direct Training for Spiking Neural Networks: Faster, Larger, Better.” In *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1311–18.

Xiao, Han, Kashif Rasul, and Roland Vollgraf. 2017. “Fashion-Mnist: A Novel Image Dataset for Benchmarking Machine Learning Algorithms.” *ArXiv Preprint ArXiv:1708.07747*.

Yanguas-Gil, Angel. 2020. “Coarse Scale Representation of Spiking Neural Networks: Backpropagation through Spikes and Application to Neuromorphic Hardware.” In *International Conference on Neuromorphic Systems 2020*, 1–7.

Zeng, Guanxiong, Yang Chen, Bo Cui, and Shan Yu. 2019. “Continual Learning of Context-Dependent Processing in Neural Networks.” *Nature Machine Intelligence* 1 (8): 364–72.

Zenke, Friedemann, Ben Poole, and Surya Ganguli. 2017. “Continual Learning through Synaptic Intelligence.” In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3987–95. JMLR. org.

Zhang, Lei, Shengyuan Zhou, Tian Zhi, Zidong Du, and Yunji Chen. 2019. “Tdsnn: From Deep Neural Networks to Deep Spike Neural Networks with Temporal-Coding.” In *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:1319–26.