Differentiable plasticity: training plastic neural networks with backpropagation

Thomas Miconi, Jeff Clune, Kenneth O. Stanley

MACHINE INTELLIGENCE COMMUNITY

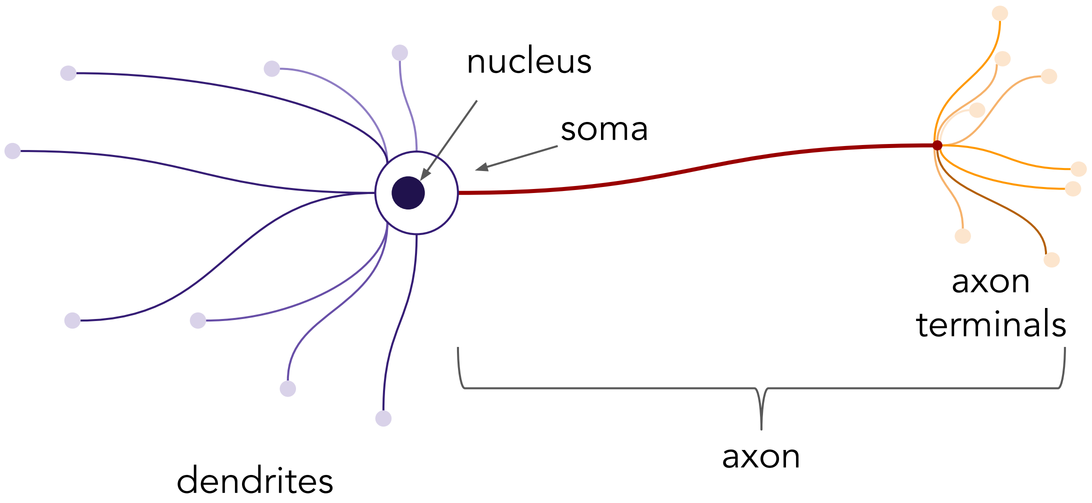
Justin Chen
May 23th, 2018

Motivation

1. Current architectures and training methods (GD and BP) do not support continuously learning new tasks - **catastrophic forgetting**
2. Biological neural networks
   a. Able to continuously integrate new information without forgetting
   b. Obtain information for adaptive behavior from environment
3. Goal:
   a. **Adaptive connections** using Hebbian Plasticity
   b. Use **backpropagation** to train **weights and plasticity**

MIC

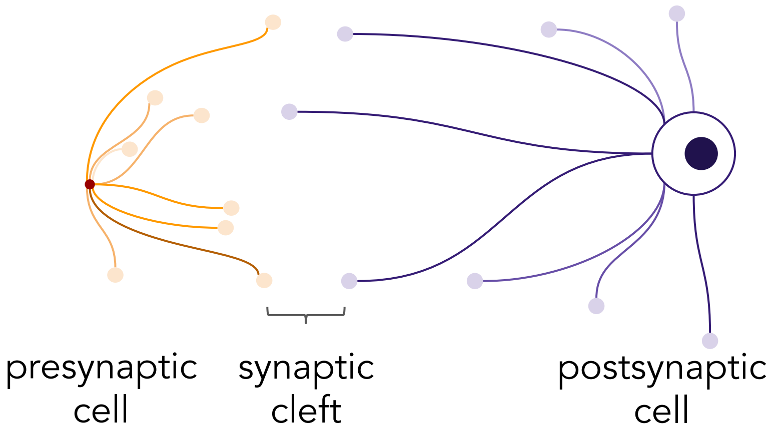
Biological Neuron

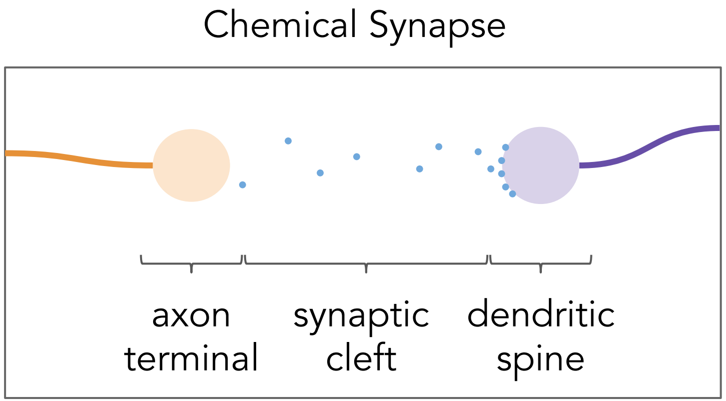


Electrical messages travel from dendrites to axon terminals

Soma processes information based on incoming signal

Biological Synapse





Neurotransmitters emitted by axon terminal across synaptic cleft to dendritic spine which sends electrical signal to soma

Soma measures Excitatory Postsynaptic Potential (EPSP) and Inhibitory Postsynaptic Potential (IPSP) signals

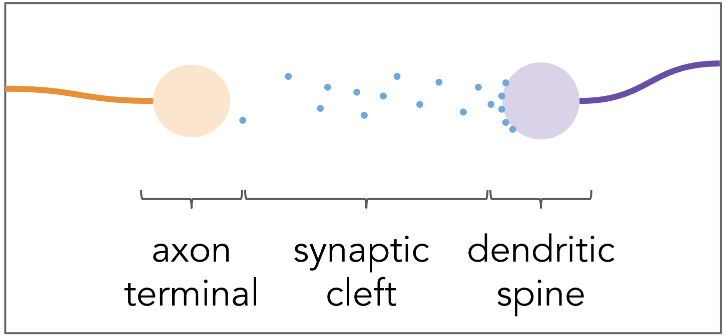
Strength of EPSP, measured by amplitude, **can change over time**

Learning in Biological Neurons

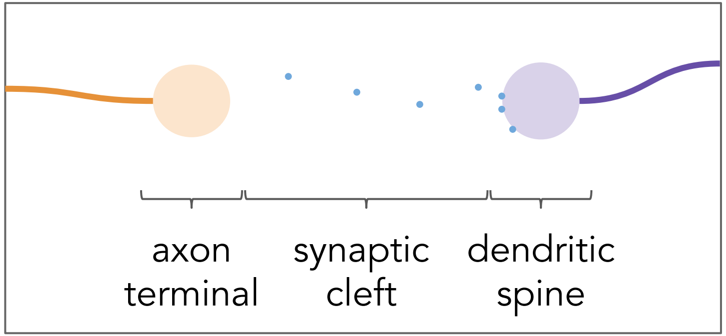
Synaptic plasticity - Changes in synapses

Cortex adapts with more neurons responding to more frequent and prominent stimuli and responding less to infrequent and nonsalient stimuli

More activity



Less activity



Hebbian Learning Theory

- **Hebbian Principle** - Cells that cause each other to activate develop stronger connections with each other

“When an axon of cell A is near enough to excite cell B or repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A’s efficiency, as one of the cells firing B, is increased."

Hebbian Learning Properties

1. **Causality** - Presynaptic cell **causes** firing (activation) of post-synaptic cell
2. **Locality** - Only **local information** at synapse updates response strength
3. **Correlation** - **Both cells** must be **active**, therefore, **learning rule** must be sensitive to correlations between the **two neurons**
4. **Positive Feedback Loop** - Correlation between input and output increases connection weight, which increase correlation of activation

$$\underbrace{\Delta w_i}_{\text{Change in the individual } i^{th} \text{ weight } w} = \underbrace{\eta}_{\text{Learning rate}} \underbrace{x_i}_{i^{th} \text{ input from post-synaptic cell}} \underbrace{y}_{\text{Postsynaptic response}}$$

Causality, Locality, Correlation

$$\Delta w_i = \eta x_i y$$

Pre-synaptic Cell \ Post-synaptic Cell	Low	High
Low	+	-
High	-	+

Correlation

Hebbian Trace

- **Hebbian Trace** - **Exponential moving average** of pre- and post-synaptic activities **per individual connection** independent of weights
  - Want to consider **all previous** timesteps
  - Captures EMA of **correlation** between **two neurons**
  - Past observations should **never** be **weighted zero**

Hebbian Trace

$$\underbrace{Hebb_k(t)}_{\text{Hebbian trace of } k^{th} \text{ incoming connection at timestep } t} = \underbrace{(1-\gamma)}_{\text{Proportion of previous trace to remember}} * \underbrace{Hebb_k(t-1)}_{\text{Trace at previous timestep}} + \underbrace{\underbrace{\gamma}_{\text{Time constant}} * \overbrace{\underbrace{x_k(t)}_{\text{Pre-synaptic cell}} * \underbrace{y(t)}_{\text{Post-synaptic cell}}}^{\text{Correlation between cells}}}_{\text{Discounted correlation at current timestep}}$$

$$\gamma \in [0,1]$$

$$Hebb_k(t) \in \mathrm{R}$$

Post-synaptic Cell Response

$$y(t) = tanh\left\{ \sum_{k \in inputs} [\overbrace{w_k}^{\text{Fixed weight}} \underbrace{x_k(t)}_{\text{Pre-synaptic cell}} + \overbrace{\alpha_k}^{\text{Plastic parameter}} \underbrace{Hebb_k(t)}_{\text{Connection-specific time-dependent quantity}} \overbrace{x_k(t)}^{\text{Pre-synaptic cell}}] + \underbrace{b}_{\text{Bias parameter}} \right\}$$

$$tanh \in (-1, 1)$$

Post-synaptic Neuron Response Vector Notation

$$y = tanh\left(\left(\sum_{k \in inputs} x_k(t)[w_k + \alpha_k Hebb_k(t)]\right) + b\right)$$

$$z_k(t) = w_k + \alpha_k Hebb_k(t)$$

$$= tanh\left(\left(\sum_{k \in inputs} x_k(t) z_k(t)\right) + b\right)$$

Post-synaptic Layer Response Matrix Notation

$$= \bar{z}^{\top}(t)\bar{x}(t) + \bar{b}$$

$$z = \begin{bmatrix} (w_{11}\alpha_{11}Hebb_{11}(t))+ & \dots & +(w_{1l}\alpha_{1l}Hebb_{1l}(t)) + b_{1l+1} \\ \vdots & \ddots & \vdots \\ (w_{j1}\alpha_{j1}Hebb_{j1}(t))+ & \dots & +(w_{jl}\alpha_{jl}Hebb_{jl}(t)) + b_{jl+1} \end{bmatrix} \begin{bmatrix} x_1(t) \\ \vdots \\ x_k(t) \\ \vdots \\ 1 \end{bmatrix}$$

$$= (w + \alpha \odot Hebb(t))x(t)$$

$$y^{(i)} = tanh(z)$$

Activation of i^{th} layer

Gradients of Connection Strength

Connection strength (activity of **post-synaptic** cell) depends on **all past activity**

w_k Determines connection's baseline value

a_k Specifies influence of Hebbian trace on k^{th} connection

$$\frac{\partial y(t_z)}{\partial w_k} = \left(1 - tanh^2(y)\right)\left(x_k(t_z)Hebb_k(t_z) + \sum_{l \in inputs} [w_l x_l(t_z) \sum_{t_u < t_z} (1-\gamma)\gamma^{t_z - t_u} x_l(t_u) \frac{\partial y(t_u)}{\partial w_k}]\right)$$

$$\frac{\partial y(t_z)}{\partial \alpha_k} = \left(1 - tanh^2(y)\right)\left(x_k(t_z)Hebb_k(t_z) + \sum_{l \in inputs} [\alpha_l x_l(t_z) \sum_{t_u < t_z} (1-\gamma)\gamma^{t_z - t_u} x_l(t_u) \frac{\partial y(t_u)}{\partial \alpha_k}]\right)$$

Gradient of Response w.r.t. Fixed Weight

$$\frac{\partial y(t_z)}{\partial w_k} = \left(1 - tanh^2(y)\right)\left(x_k(t_z)Hebb_k(t_z) + \underbrace{\sum_{l \in inputs}}_{\text{Account for } \textbf{all inputs} \text{ to } y} [w_l x_l(t_z) \underbrace{\sum_{t_u < t_z} (1-\gamma)\gamma^{t_z - t_u} x_l(t_u) \frac{\partial y(t_u)}{\partial w_k}]}_{\text{Partial derivative of the Hebbian traces at time } t_z \text{ w.r.t. } w_k}\right)$$

Gradient of Response w.r.t. Plastic Parameter

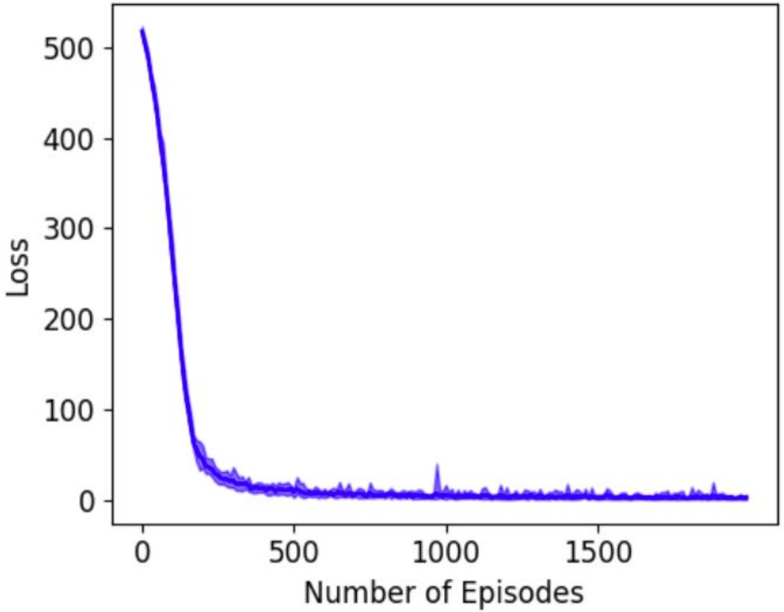
$$\frac{\partial y(t_z)}{\partial \alpha_k} = \Big(1 - tanh^2(y)\Big)\Big(x_k(t_z)Hebb_k(t_z) + \underbrace{\sum_{l \in inputs}}_{\text{Account for all inputs to } y} [\alpha_l x_l(t_z) \underbrace{\sum_{t_u < t_z} (1-\gamma)\gamma^{t_z - t_u} x_l(t_u) \frac{\partial y(t_u)}{\partial \alpha_k}]}_{\text{Partial derivative of the Hebbian traces at time } t_z \text{ w.r.t. } a_k}\Big)$$

Pattern Memorization, One-shot, Reinforcement Learning

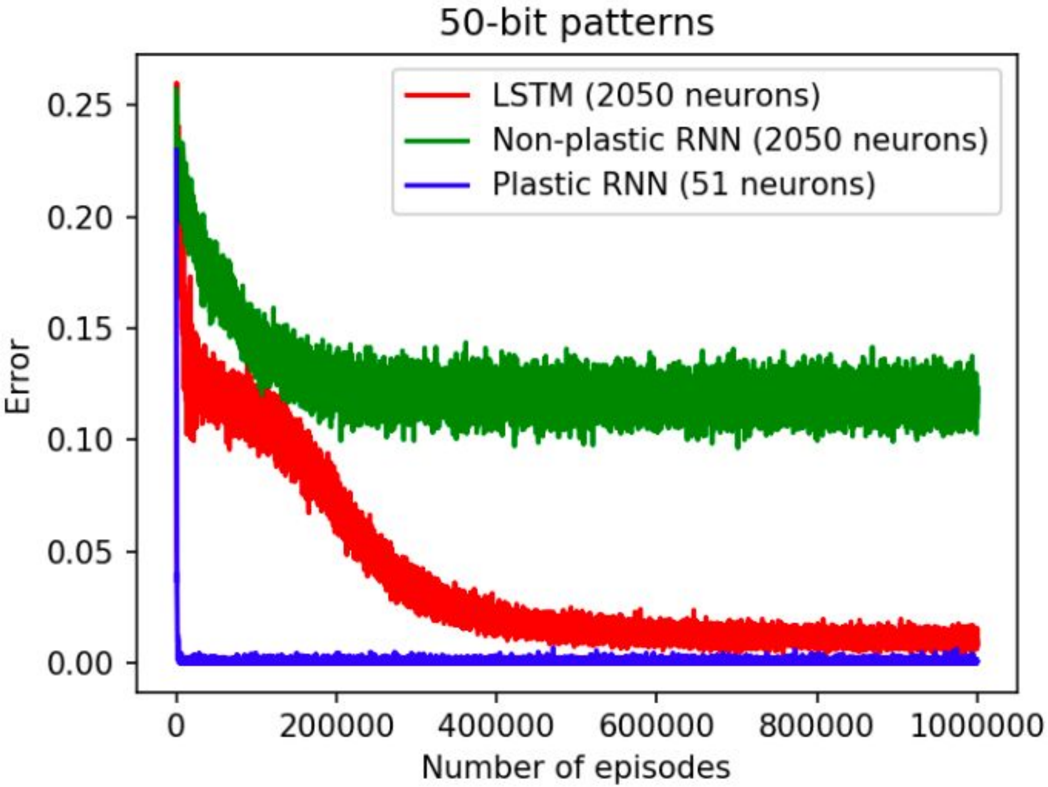
Experiments and Results

Binary Pattern Memorization

- Goal: **Content-addressable memory/Auto-association** - quickly memorizing high-dimensional patterns and reconstructing partially degraded versions
- Architecture: 1000-neuron RNN
- Input: 1000-dimensional binary vectors
- Loss: L2

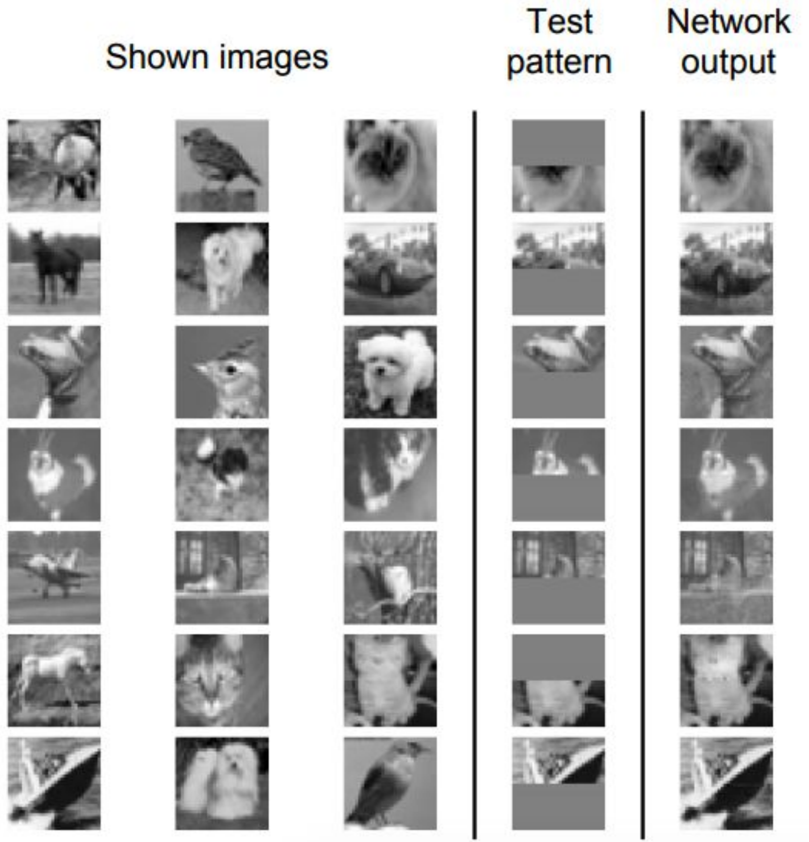


Comparison with Non-Plastic RNN

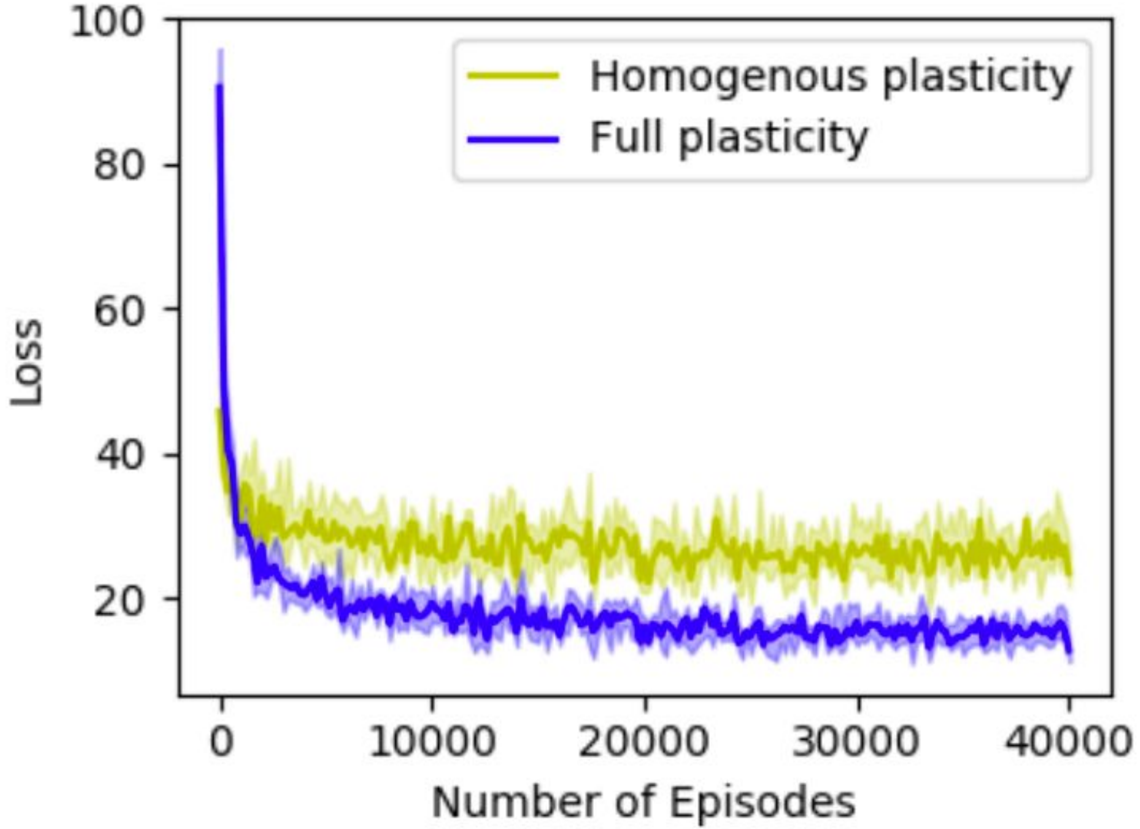


Natural Image Memorization

- Goal: Reconstruct the original image
- Architecture: 1025-neuron RNN
- Input: Half image
- Loss: L2



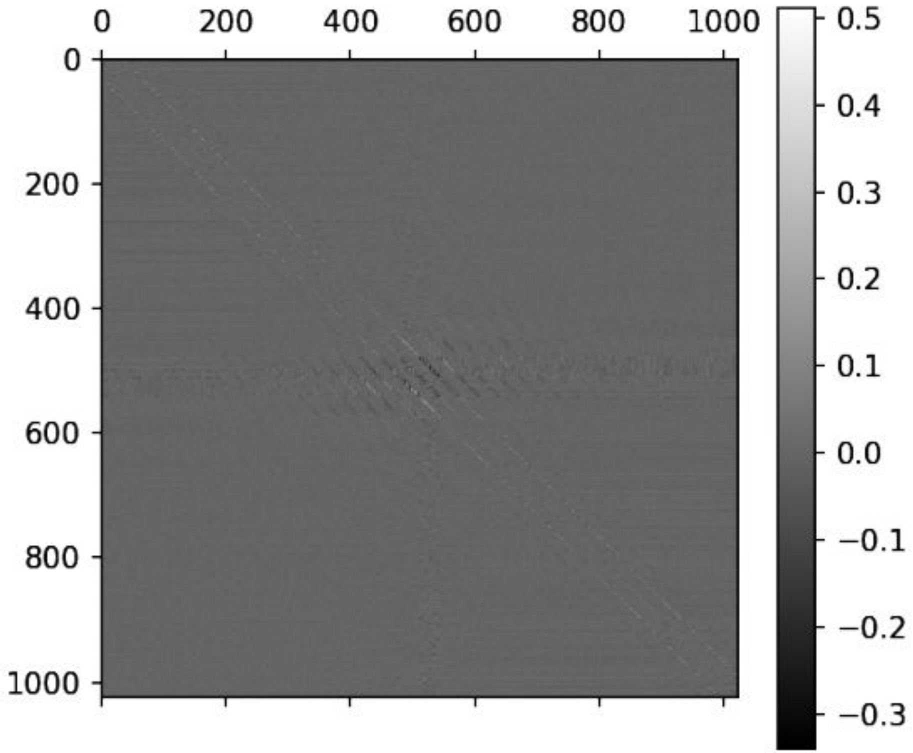
Natural Image Memorization



Natural Image Memorization Baseline Weights

Each column is input to a single cell

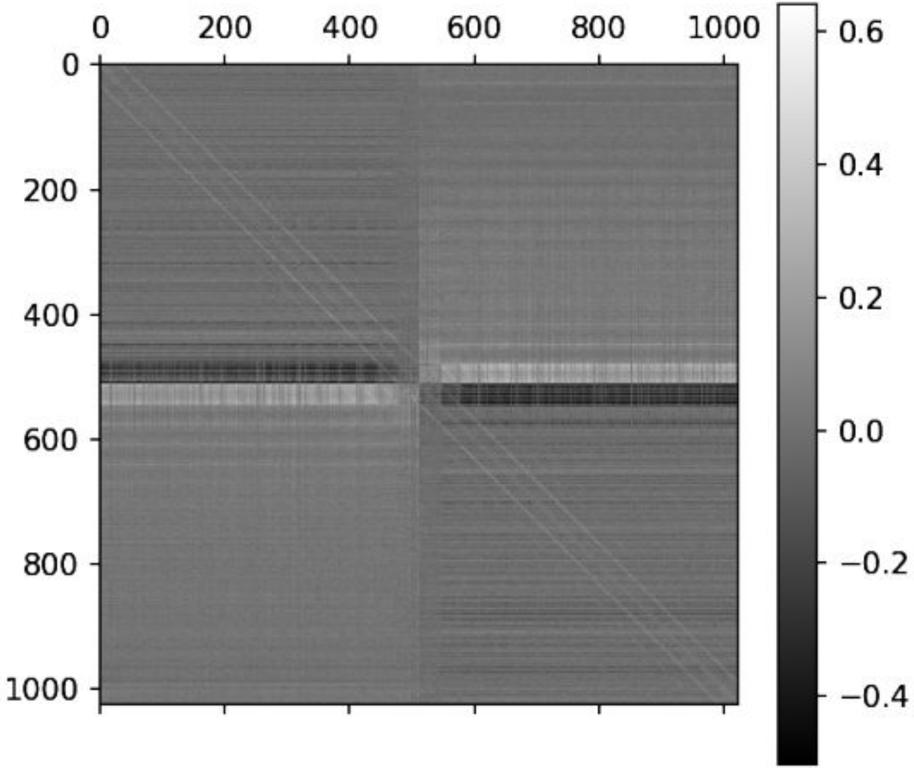
Vertically adjacent entries describe inputs from horizontally adjacent pixels



Natural Image Memorization Plastic Coefficients

Each column is input to a single cell

Vertically adjacent entries describe inputs from horizontally adjacent pixels

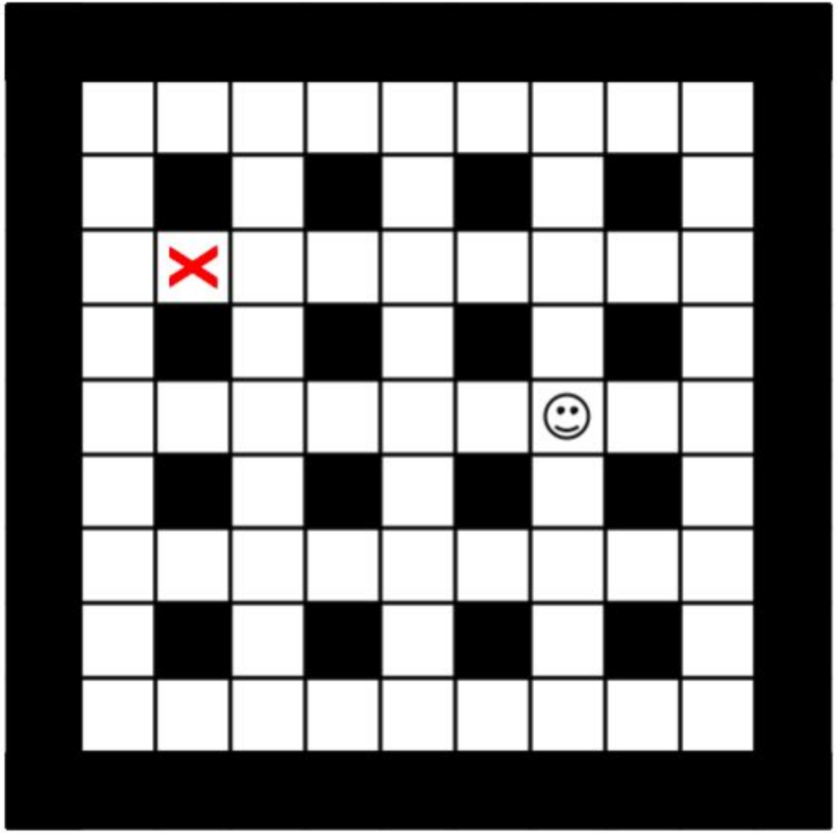


Omniglot One-Shot Classification

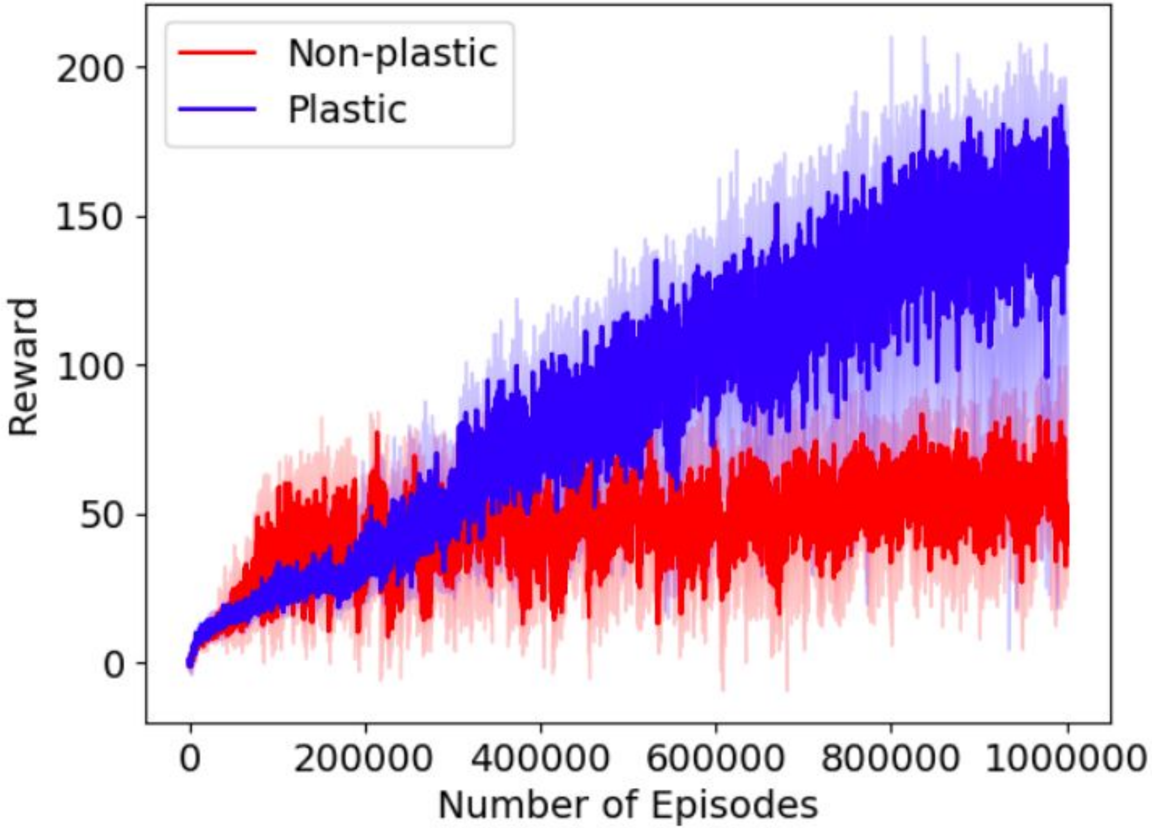
Vinyals et al. (Matching Networks) (Vinyals et al., 2016)	Snell et al. (ProtoNets) (Snell et al., 2017)	Finn et al. (MAML) (Finn et al., 2017)	Mishra et al. (SNAIL) (Mishra et al., 2017)	DP (Ours)
98.1 %	97.4%	98.7% $\pm$ 0.4%	99.07% $\pm$ 0.16	98.5% $\pm$ 0.57

Maze Exploration

- Goal: Find reward (red X)
- Architecture: 100-neuron RNN
- Input: Binary vector of agent's location



Maze Exploration



Learning to learn with backpropagation of Hebbian plasticity

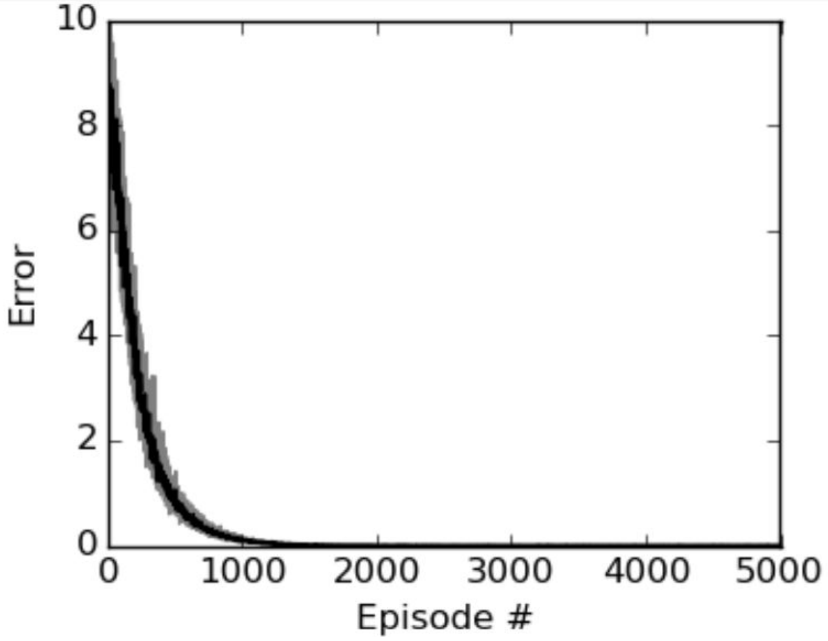
Experiments

Experiments

- All Hebbian traces are **initialized to 0** at the beginning of each episode
- Tasks:
  - Pattern Completion
  - One-shot Learning of Arbitrary Patterns
  - Reversal Learning

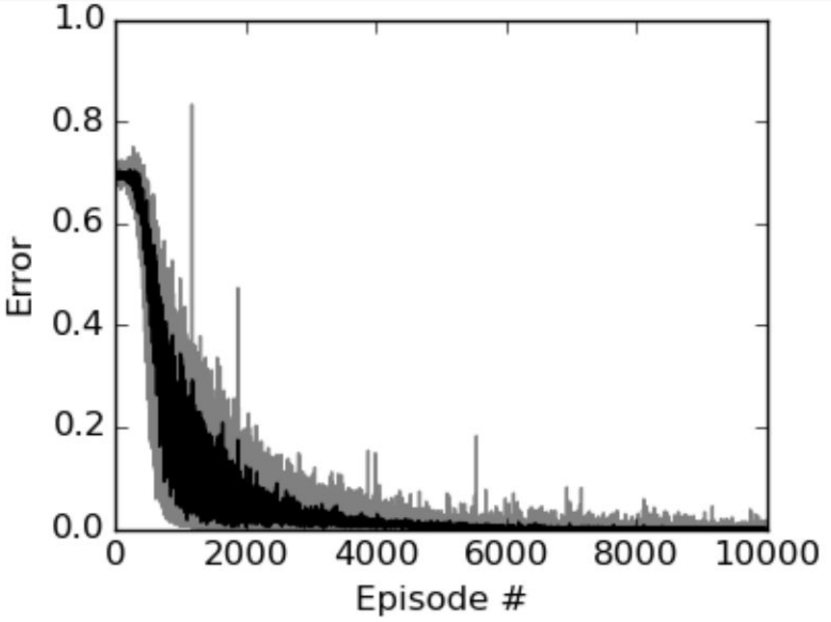
Pattern Completion

- Goal: Given a vector with only one non-zero bit, generate the full vector
- Architecture: single-layer NxN network
- Input: Random binary vector with at least one non-zero bit
- Loss: Manhattan distance



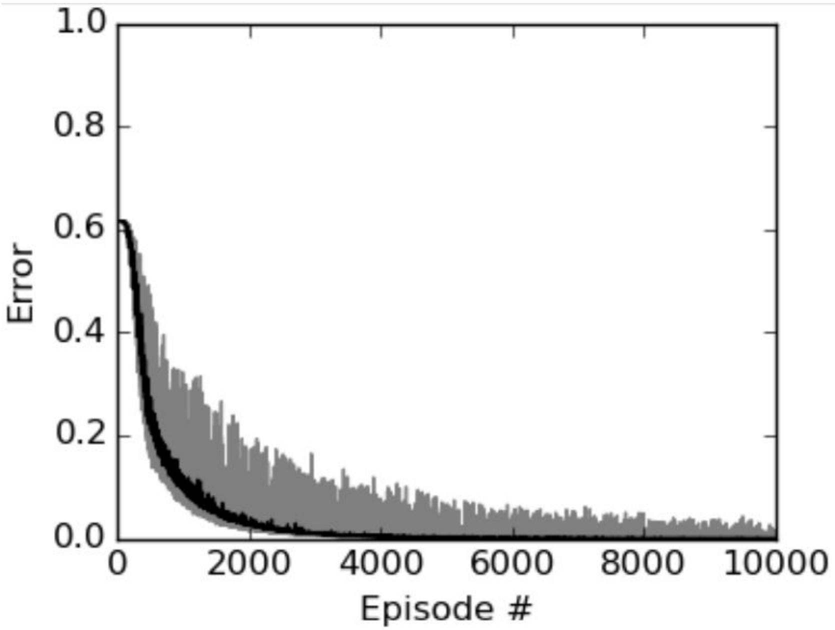
One-shot Learning of Arbitrary Patterns

- Goal: Label random two binary vectors
- Architecture: N, 2, 2 network with plasticity only in the first layer
- Input: Random binary vectors with at least one non-zero bit
- Loss: Cross-entropy



Reversal Learning

- Goal: Continual learning by switching labels
- Architecture: N, 2, 2 network with negative plasticity only in the first layer
    - **Negative plasticity** - Activation is anti-correlated in presence of learned stimuli
    - Decreases Hebbian trace over time
- Input: Random binary vector with at least one non-zero bit
- Loss: Manhattan distance



Citations and Further Reading

1. Miconi, Thomas. "**Learning to learn with backpropagation of Hebbian plasticity**." arXiv preprint arXiv:1609.02228 (2016).
2. Miconi, Thomas, Jeff Clune, and Kenneth O. Stanley. "**Differentiable plasticity: training plastic neural networks with backpropagation**." arXiv preprint arXiv:1804.02464 (2018).
3. Hebb, Donald Olding. **The organization of behavior: A neuropsychological theory**. Psychology Press, 1949.
4. Gerstner, Wulfram. "**Hebbian learning and plasticity**." From neuron to cognition via computational neuroscience (2011): 0-25.