University of BRISTOL

Supervised learning in recurrent neural networks

Seán Froudist-Walsh

Lecturer in Computational Neuroscience

1

1

Intended Learning Outcomes

- By the end of this video you will be able to:
  - understand backpropagation through time, and its strengths and limitations in terms of performance and biological realism
  - build and write equations for distinct RNN architectures and detail how they deal with common issues to training RNNs over long timescales
  - describe the main ideas behind two biologically-inspired learning rules for RNNs
  - describe the potential benefits of, and best practices for, applying RNNs to understanding the neural mechanisms of cognition

2

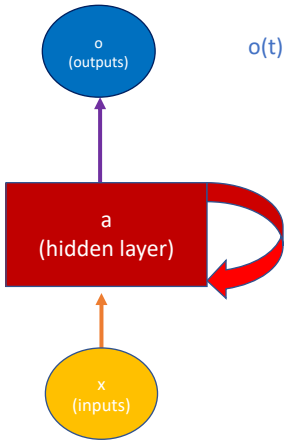
2

Recurrent neural networks (RNNs) in AI

- The basic (”vanilla”) RNN

$a(t) = f(W_{x\to a}\, x(t) + W_{a\to a}\, a(t-1) + b_1)$

$o(t) = g(W_{a\to o}\, a(t) + b_2)$



The hidden layer activity
is a nonlinear function (e.g. ReLU, sigmoid)
of a weighted sum of the inputs
and a weighted sum
of the hidden layer activity from one timestep ago
plus a bias

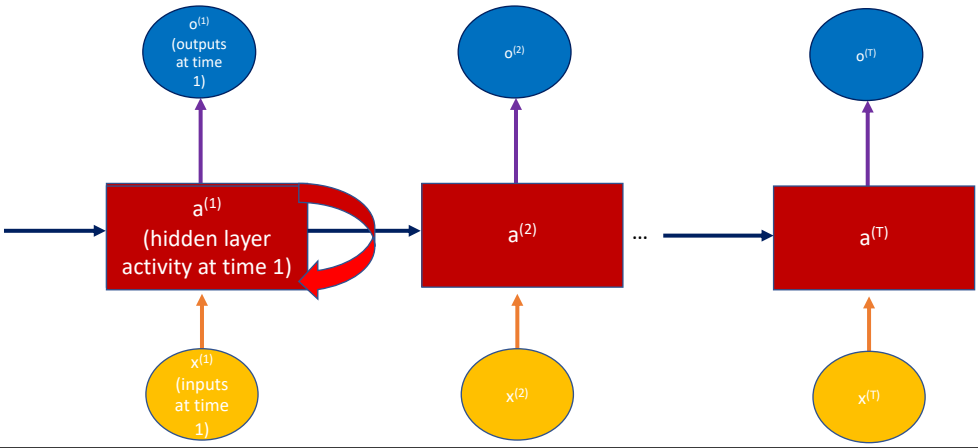
The output layer activity is a nonlinear function
is a weighted sum
of the hidden layer activity
plus a bias

Elman, *Cog. Sci.*, 1990

3

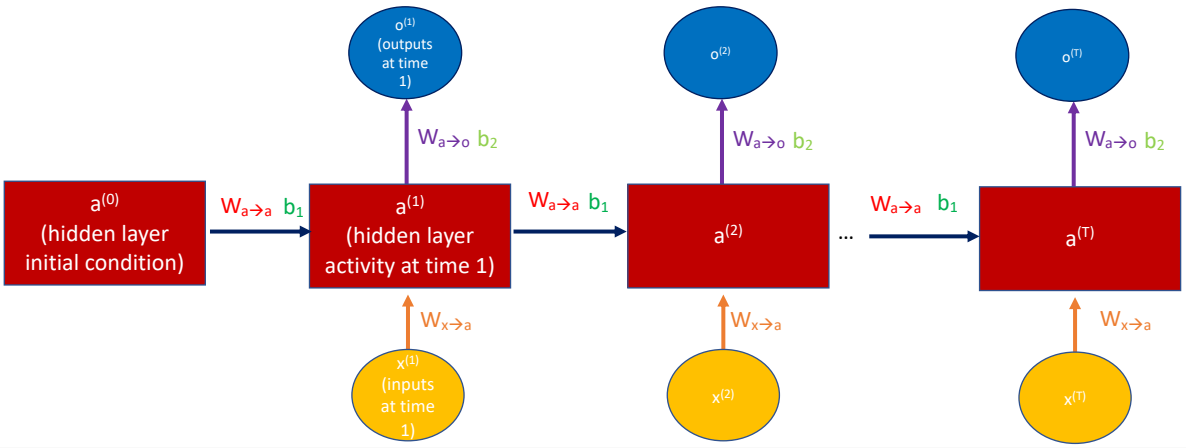
3

Unrolled RNN



4

Forward propagation in the RNN



5

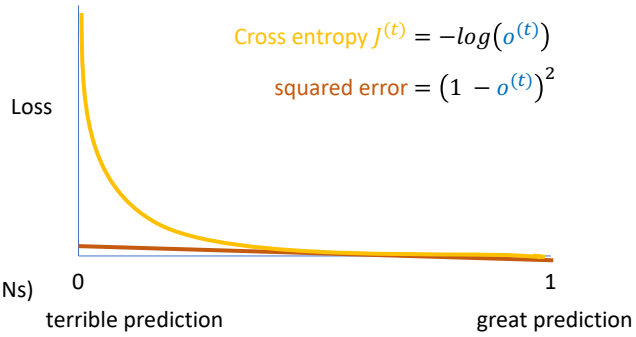
Cross entropy loss

Loss function (Cross entropy)

$$J^{(t)} = -y^{(t)} log\left(o^{(t)}\right)$$

$$J = \sum_{t=1}^{T} J^{(t)}$$

Used for classification tasks (not specific to RNNs)



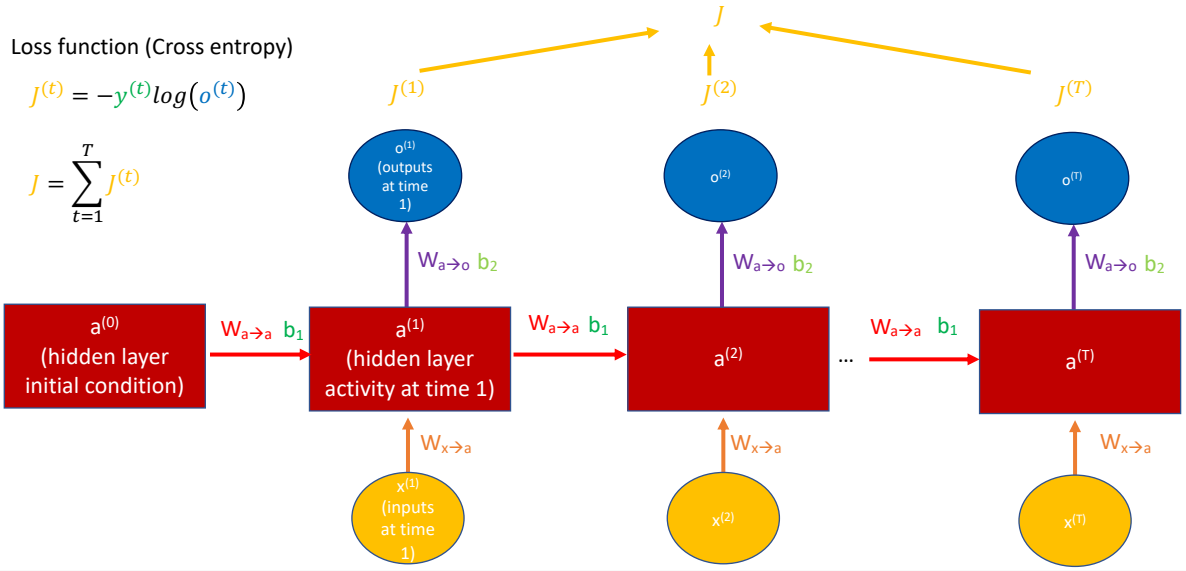
6

Loss in the RNN

Loss function (Cross entropy)

$$J^{(t)} = -y^{(t)} log\left(o^{(t)}\right)$$

$$J = \sum_{t=1}^{T} J^{(t)}$$



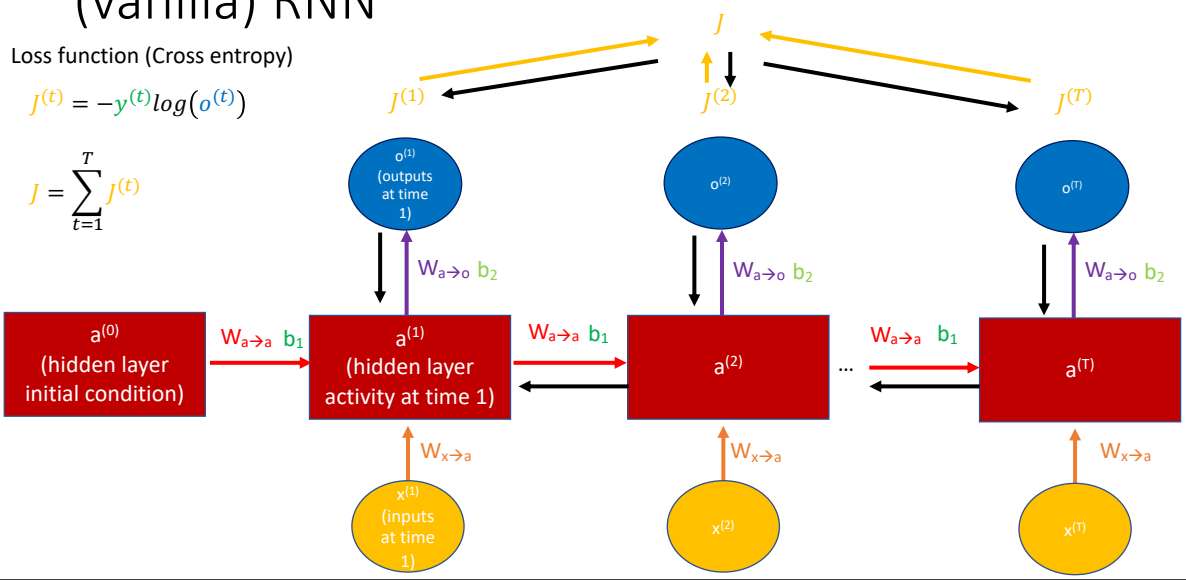
7

Backpropagation in the (vanilla) RNN

Loss function (Cross entropy)

$$J^{(t)} = -y^{(t)} log\left(o^{(t)}\right)$$

$$J = \sum_{t=1}^{T} J^{(t)}$$



8

Backpropagation through time (BPTT)

Exploding gradient problem

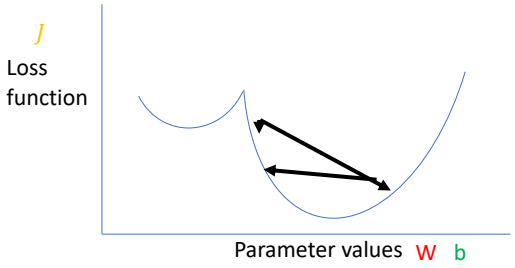
For 100 timesteps, w = 2

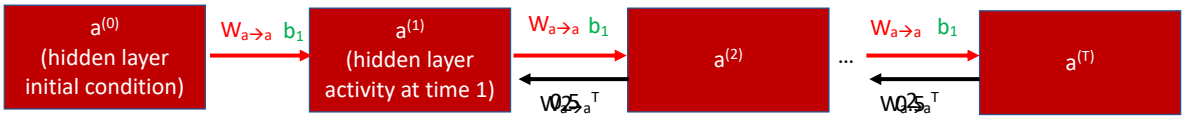
2^{100}= 1,267,650,600,228,229,401,496,703,205,376

Vanishing gradient problem

For 100 timesteps, w = 0.5

0.5^{100}= 0.000000000000000000000000000008





For a weight matrix $W_{a \to a}$

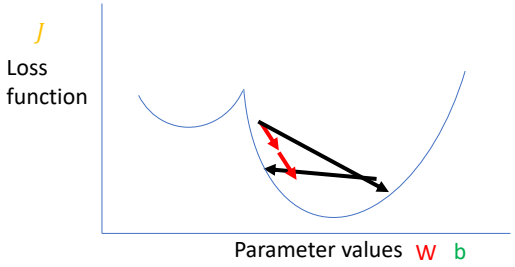
if the largest eigenvalue > 1 *Exploding gradient problem*

if the largest eigenvalue < 1 *Vanishing gradient problem*

9

Fixing exploding gradients – Gradient clipping

- *"When the traditional gradient descent algorithm proposes to make a very large step, the gradient clipping heuristic intervenes to reduce the step size to be small enough that it is less likely to go outside the region where the gradient indicates the direction of approximately steepest descent"*



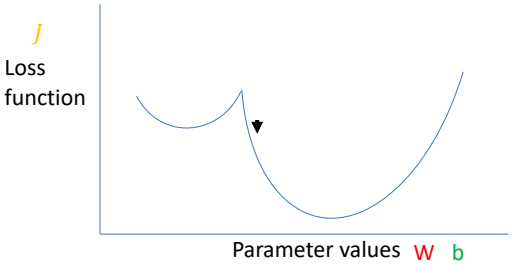
Pascanu et al., *ICML*, 2013
Goodfellow et al., *Deep Learning*, 2016

10

Understanding the vanishing gradient problem

The vanishing gradient problem for vanilla RNNs means that the algorithm believes that activity early timepoints will not affect the loss.

Therefore it will not make big adjustments to the weights based on activity at early timepoints.

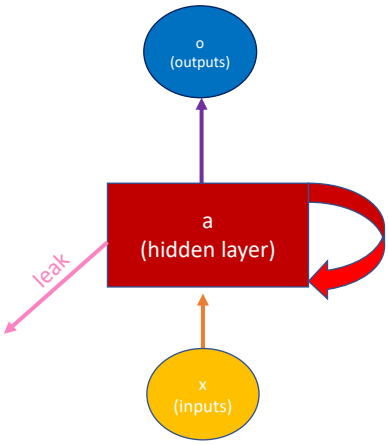


This isn't because early time steps don't influence the loss,

but because the backpropagation-through-time process struggles to recognise and act upon their influence.

11

Leaky RNNs for neuro-AI



Discrete time version (Euler method - written in code)

$$a(t) = a(t-1) + \frac{\Delta t}{\tau}\left(\text{-a(t-1)} + \text{f}\left(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1\right)\right) + \text{noise}$$

$$Let\ \frac{\Delta t}{\tau} = \beta$$

$$a(t) = a(t-1) + \beta\left(\text{-a(t-1)} + \text{f}\left(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1\right)\right)$$

$$a(t) = (1-\beta)\,a(t-1) + \beta\left(\text{f}\left(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1\right)\right)$$

$$new\ a = (1-\beta)\,old\ a + \beta(new\ inputs)$$

$if\ \beta = 1,$ we get back the vanilla RNN

12

Slowly vanishing gradients in leaky RNNs

$$a(t) = (1-\beta)\,a(t-1) + \beta\,\left(f(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1)\right)$$

$$\frac{\delta J}{\delta a(t-k)} = \frac{\delta J}{\delta a(t)}\frac{\delta a(t)}{\delta a(t-1)}\frac{\delta a(t-1)}{\delta a(t-2)}\ldots\frac{\delta a(t-k+1)}{\delta a(t-k)}$$

$$\frac{\delta a(t)}{\delta a(t-1)} = (1-\beta) + \beta\, f'(\ldots)\,W_{a\to a} \qquad \frac{\delta a(t-1)}{\delta a(t-2)} = (1-\beta) + \beta\, f'(\ldots)\,W_{a\to a}$$

So the terms $(1-\beta)$ and the recurrent weights $\beta\, f'(\ldots)\, W_{a\to a}$ are multiplied over and over again as we backpropagate through time.

$if\ \beta << 1,$ then the magnitude of the gradient is dominated by $(1-\beta)$.

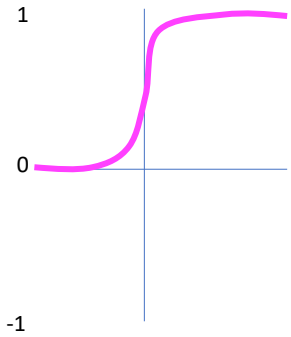
On each backpropagation step, we multiply by a number close to 1, and the gradient vanishes slowly.

This will allow us to use activity further in the past to influence training.

13

Sigmoid activation function

sigmoid $f(x) = \sigma(x) = \dfrac{e^x}{e^x + 1}$



14

Light Gated Recurrent Units (Light GRUs)

Leaky RNN – scalar β (just one number controls the timescale of remembering/updating for all units)

$a(t) = (1-\beta)\,a(t-1) + \beta\,\big(f(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1)\big)$

Light GRU – vector $\boldsymbol{\beta}$ (a different number controls the timescale for each unit, and this changes in time)

$a(t) = (1-\boldsymbol{\beta}(t)) \circ a(t-1) + \boldsymbol{\beta} \circ \big(f(W_{x\to a}\,x(t) + W_{a\to a}\,a(t\text{-}1) + b_1)\big)$

The gate $\boldsymbol{\beta}$ is learned, and depends on the inputs and recurrent connections

$\boldsymbol{\beta} = \sigma(W_{x\to\beta}\,x(t) + W_{a\to\beta}\,a(t\text{-}1) + b_1)$

f = ReLU

The gate controls the fraction (sigmoid) of newly calculated activity that will contribute to the next memory state $a(t)$ for each unit

normalise activations across units in a layer (subtract mean & divide by standard deviation plus a small number)

something (superficially) similar happens in the brain

Ravanelli et al., *IEEE*, 2018

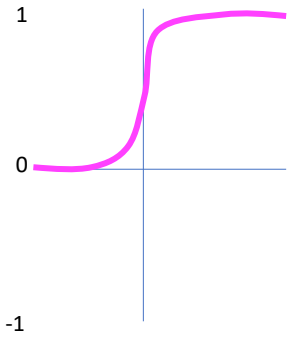
Carandini & Heeger, *Nat. Rev. Neurosci., 2012*

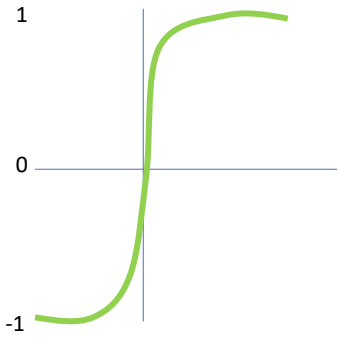
15

Sigmoid & tanh activation functions

sigmoid $f(x) = \sigma(x) = \dfrac{e^x}{e^x + 1}$

tanh $f(x) = \tanh(x) = \dfrac{e^x - e^{-x}}{e^x + e^{-x}}$





16

Gated Recurrent Units (GRUs)

Light GRU

$a(t) = (1 - \boldsymbol{\beta}(t)) \circ a(t-1) + \boldsymbol{\beta} \circ \left(f(W_{x\to a}\, x(t) + W_{a\to a}\, a(t\text{-}1) + b_1) \right)$

GRU (a separate gate on previous timestep activations affects the update)

$a(t) = (1 - \boldsymbol{\beta}(t)) \circ a(t-1) + \boldsymbol{\beta} \circ \left(f(W_{x\to a}\, x(t) + W_{a\to a}(a(t\text{-}1) \circ \gamma) + b_1) \right)$

The gates $\boldsymbol{\beta}$ and γ are learned, and depend on the inputs and recurrent activity/weights

$\boldsymbol{\beta} = \sigma(W_{x\to\beta} x(t) + W_{a\to\beta} a(t\text{-}1) + b_\beta)$

$\gamma = \sigma(W_{x\to\gamma} x(t) + W_{a\to\gamma} a(t\text{-}1) + b_\gamma)$

f = tanh

Cho et al., *arXiv,* 2014

17

LSTM (Long Short-Term Memory)

LSTMs maintain and update separate long-term memory and short-term memory variables.

They control what how long and short-term memories are updated by use of 3 gates.

$every\ gate = \sigma(z(t)) = \sigma(W_h h(t-1) + W_x \boldsymbol{x}(t) + b)$ depends on the short-term memory, inputs and a bias

$\boldsymbol{f(t)} = \sigma(W_{h\to f}\boldsymbol{h}(t-1) + W_{x\to f}\boldsymbol{x}(t) + \boldsymbol{b_f})$

$\boldsymbol{i(t)} = \sigma(W_{h\to i}\boldsymbol{h}(t-1) + W_{x\to i}\boldsymbol{x}(t) + \boldsymbol{b_i})$

each gate controls the fraction (sigmoid) of 'something' contributing to a new memory state

$\boldsymbol{o(t)} = \sigma(W_{h\to o}\boldsymbol{h}(t-1) + W_{x\to o}\boldsymbol{x}(t) + \boldsymbol{b_o})$

each candidate new memory uses a tanh activation function

$\boldsymbol{c}(t) = \boldsymbol{f(t)} \circ \boldsymbol{c}(t-1) + \boldsymbol{i(t)} \circ \hat{\boldsymbol{c}}(t)$ $\quad \hat{c}(t) = \tanh(z_c(t))$ $\quad z_c = (W_{h\to c}\boldsymbol{h}(t-1) + W_{x\to c}\boldsymbol{x}(t) + \boldsymbol{b_c})$

The new long-term memory is the fraction you don't forget of the old long-term memory plus the fraction you allow in of the candidate new long-term memory.

Note: 'long-term memory' here is not equivalent to long-term memory in neuroscience/psychology, which can die off in activity and be recalled years later.

$h(t) = \boldsymbol{o(t)} \circ \hat{\boldsymbol{h}}(t)$ $\quad \hat{\boldsymbol{h}}(t) = \tanh(c(t))$

The new short-term memory (output) is the fraction you output from the candidate new short-term memory.

Hochreiter & Schmidhuber, *Neural Computation,* 1997

18

Solving vanishing gradients with LSTMs

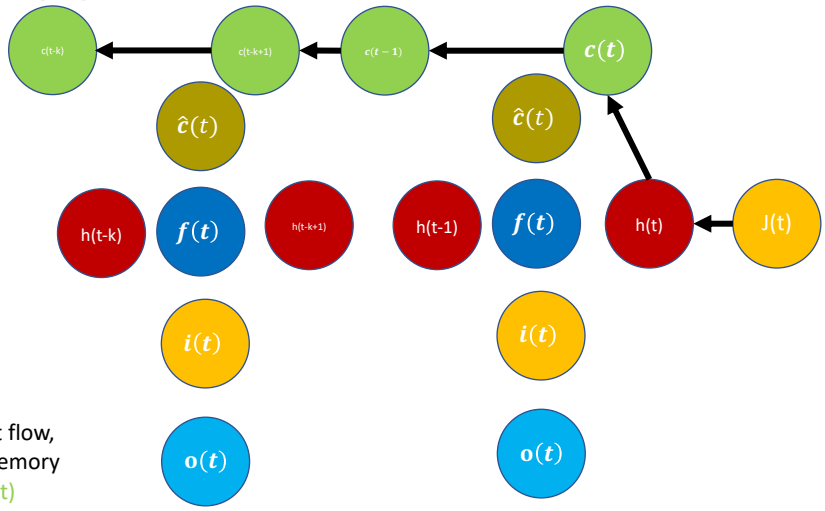
To avoid vanishing gradients, we just need at least one of the paths back from the loss to not vanish.
This will ensure a steady flow of the gradient back through time.

$$\frac{\partial J(t)}{\partial c(t-k)} = \frac{\partial J(t)}{\partial h(t)} \frac{\partial c(t)}{\partial c(t-1)} \frac{\partial c(t-1)}{\partial c(t-2)} \ldots \frac{\partial c(t-k+1)}{\partial c(t-k)}$$

$$\boldsymbol{c}(t) = \boldsymbol{f}(\boldsymbol{t}) \circ \boldsymbol{c}(t-1) + \boldsymbol{i}(\boldsymbol{t}) \circ \hat{\boldsymbol{c}}(t)$$

$$\frac{\partial c(t)}{\partial c(t-1)} = \boldsymbol{f}(\boldsymbol{t}) + \text{stuff}$$

The forget gates thus regulate the gradient flow, according to the how much a long-term memory state c(t-1) contributed to the next state c(t)

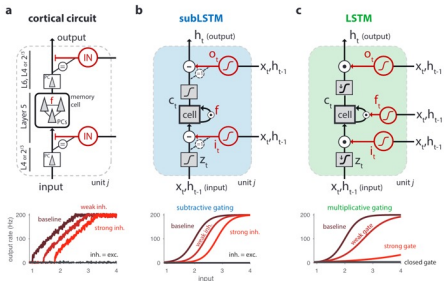


Mitesh Khapra

19

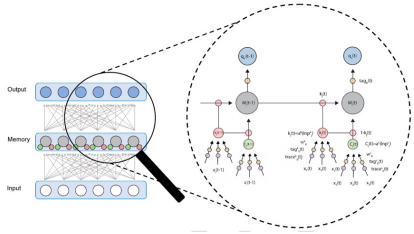
Realism in RNN architectures

- Leaky RNNs are close to traditional (untrained) recurrent neural network models in computational neuroscience
- Leaky RNNs are often observed to have brain-like dynamics
- This is not always true for other RNN architectures (vanilla RNN, light GRU, GRU, LSTM)
- However other models (light GRU, GRU, LSTM) are more effective at learning long-term dependencies
- Efforts have been made to develop more biologically-plausible versions of these more powerful architectures



biologically-inspired variant of LSTM

Costa et al., *NeurIPS*, 2017



biologically-inspired variant of light GRU

Berg et al., *bioRxiv, 2023*

20

Quiz - blackboard

- Both GRUs and LSTMs incorporate _____ mechanisms that allow them to selectively remember or forget information, which is especially crucial for sequences where important information might be spaced apart. In contrast, leaky RNNs introduce a _____ in the recurrent loop to allow the older states to decay over time.

21

Biological (im)plausibility of BPTT

1. The weights going forwards equal the weights going back (like in backpropagation for feedforward networks)
2. The same neuron must store and retrieve, with perfect accuracy, the values of its activities from all points in past

22

Biologically plausible learning in RNNs?

$$\partial W_{ab}(t) = \alpha\,[B\delta(t)]_a\, p_{ab}(t)$$

The change in weight between neurons a and b at time t depends on

the learning rate

the error distributed in proportion to random weights

and a moving average of recent co-activations between neuron a and b.

Strengths - only depends on information that each neuron should have (i.e. local)

no need re-use weights in the forward and backward directions

no need to run the activations backwards at the end of the trial to learn

Limitation - co-activations are forgotten at a rate determined by the neuronal timescale

$$p_{ab}(t) = \frac{1}{\tau}\phi'(u_a(t))h_b(t-1) + \left(1-\frac{1}{\tau}\right)p_{ab}(t-1),$$

Murray, *eLife*, 2019

$$\Delta W \propto r \cdot \sum_{t,t+1} p^t \cdot (I^{t\intercal} + I^{t+1\intercal})$$

Hook up an RNN to a variant of a Hopfield network, where memories are linked to their neighbours in time.
Cheng and Brown. *bioRxiv* (2023)

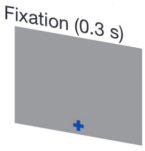
backwards replay does happen in the brain!
e.g.
Ólafsdóttir et al., *Current Biology* (2018)

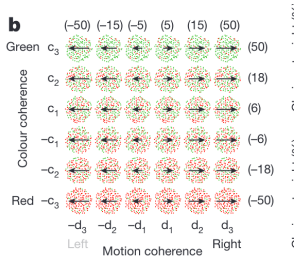
23

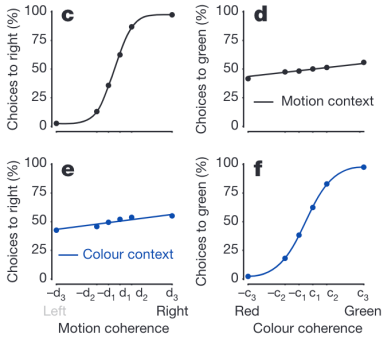
Perhaps the learning rule isn't plausible, but the learned neural network is?

24

The brain uses context to guide behaviour





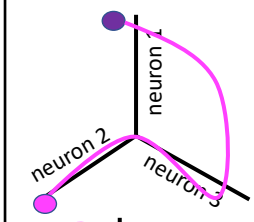


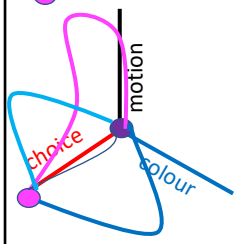
identical sensory stimuli can lead to very different behavioral responses depending on context

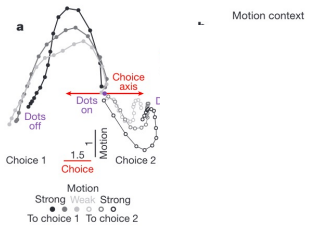
Mante*, Sussillo* et al., *Nature*, 2013

25

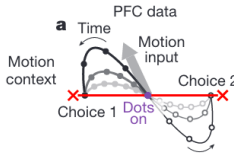
How the brain uses context to guide behaviour

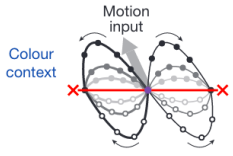






Colour information reaches prefrontal cortex but does not affect choice



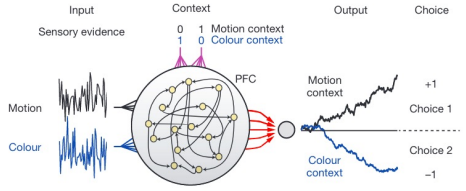


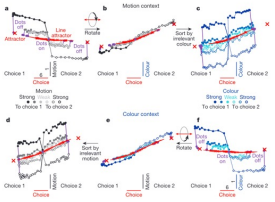
Rules out dominant "early selection" hypothesis: sensory information is 'gated out' before reaching prefrontal cortex

Mante*, Sussillo* et al., *Nature*, 2013

26

How a leaky RNN uses context to guide behaviour

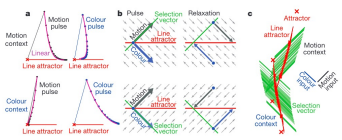




RNNs, when applied and analysed carefully, can generate new hypotheses about how the brain performs its functions

Method for using RNNs to understand neural mechanisms of cognition:

1. Train RNN on same task as individual (e.g. human, monkey mouse) performing task
2. Analyse RNN using the same methods as used to analyse real neural activity
3. If the RNN is a good match to the neural data, then perform additional 'experiments' and analyses on the RNN to enable deeper insights into neural mechanisms



This paper:

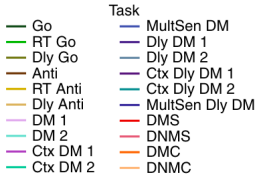
1. Leaky RNN trained with BPTT
2. The RNN activity resembled real neural population activity seen in prefrontal cortex
3. Dynamical systems analysis showed that the context modifies the 'dynamical landscape', affecting how the neural responses to motion and color guide activity towards making a choice.

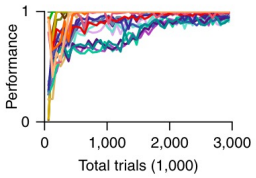
Mante*, Sussillo* et al., *Nature*, 2013

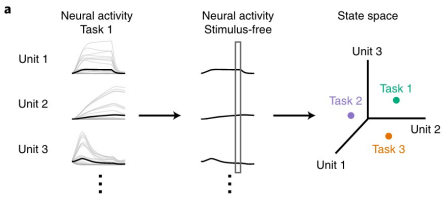
27

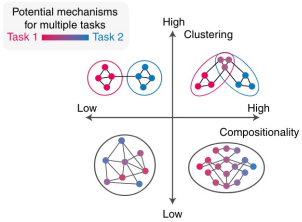
Moving beyond traditional computational neuroscience models

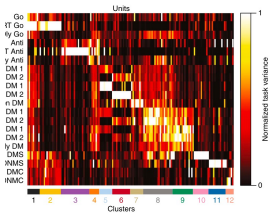
A single network performing 20 cognitive tasks

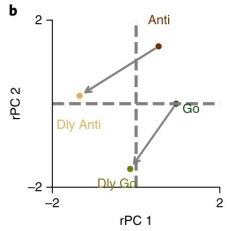












- Training neural networks can teach us how to build computational models for aspects of cognition that were previously overly complicated for 'hand-designed' networks.

Yang et al., *Nature Neuroscience*, 2019

28

Quiz - blackboard

- Even if backpropagation through time is not considered realistic, the b_____, d_____, and r_____ that emerge in trained RNNs can be, and can offer valuable insights for understanding brain function.

29

Recap

- Backpropagation through time is effective at assigning credit/blame to synapses, but is not biologically realistic
- Vanishing and exploding gradient problems stop RNNs learning from timepoints in the distant past
- Different architectures (light GRU, GRU, LSTM) tackle the vanishing gradient problem, but their biological relevance is still debated
- Biologically-realistic learning rules avoid storing all previous activity states, or retrieve them using hippocampal-replay-like mechanisms
- Even if the learning rule is not realistic, the trained network could give insights into how the brain performs cognitive tasks

30

Still curious? You can dive in deeper to any of today's topics:

- Light GRU
  - Ravanelli, Mirco, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. "Light gated recurrent units for speech recognition." *IEEE Transactions on Emerging Topics in Computational Intelligence* 2, no. 2 (2018): 92-102.
- Normalisation in the brain
  - Carandini, Matteo, and David J. Heeger. "Normalization as a canonical neural computation." *Nature Reviews Neuroscience* 13, no. 1 (2012): 51-62.
- GRU
  - Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." *arXiv preprint arXiv:1406.1078* (2014).
- LSTM
  - Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9, no. 8 (1997): 1735-1780.
  - Proof of LSTM solving the vanishing gradient problem by Mitesh Khapra
    - https://www.youtube.com/watch?v=dKKPtG-qeaI&ab_channel=NPTEL-NOCIITM
- Backpropagation through time and the brain
  - Lillicrap, Timothy P., and Adam Santoro. "Backpropagation through time and the brain." *Current opinion in neurobiology* 55 (2019): 82-89.
- Biologically reasonable alternative to backpropagation through time
  - Murray, James M. "Local online learning in recurrent networks with random feedback." *Elife* 8 (2019): e43299.
- Context-dependent decision-making in the brain and RNNs
  - Mante, Valerio, David Sussillo, Krishna V. Shenoy, and William T. Newsome. "Context-dependent computation by recurrent dynamics in prefrontal cortex." *nature* 503, no. 7474 (2013): 78-84.
- Training an RNN to perform 20 cognitive tasks
  - Yang, Guangyu Robert, Madhura R. Joglekar, H. Francis Song, William T. Newsome, and Xiao-Jing Wang. "Task representations in neural networks trained to perform many cognitive tasks." *Nature neuroscience* 22, no. 2 (2019): 297-306.

31

31