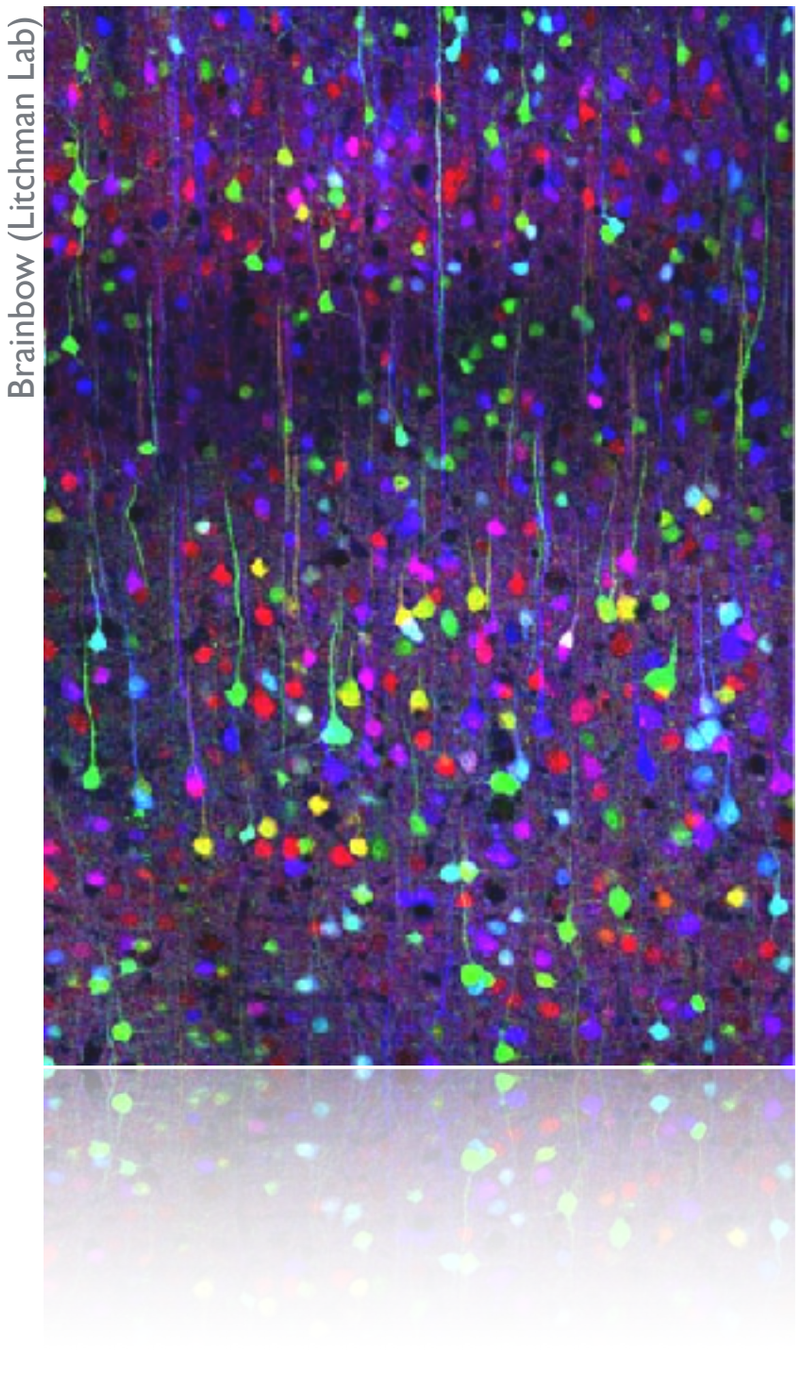
Neural Information Processing 2018/2019

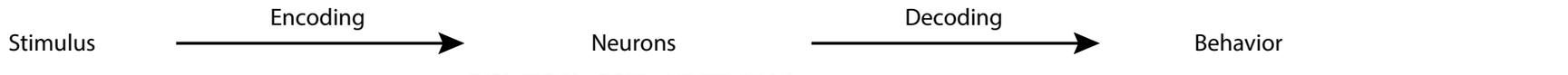
University of BRISTOL

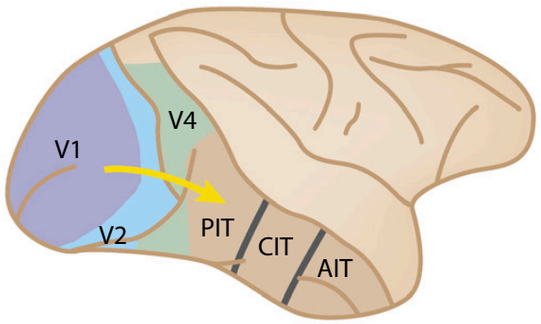

Brainbow (Litchman Lab)

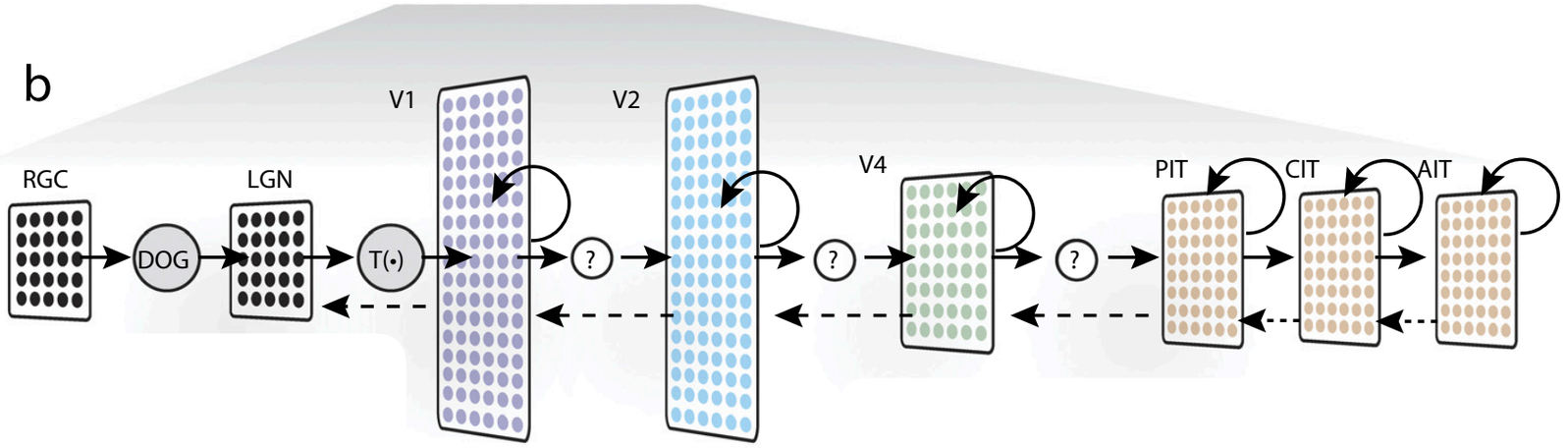
Lecture 15
Neural circuits and learning:
Temporal processing and
recurrent neural networks

Previously on *Neural Information Processing*...

Feedforward neural networks:



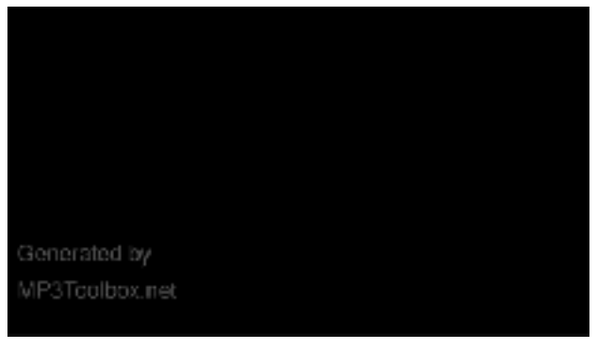




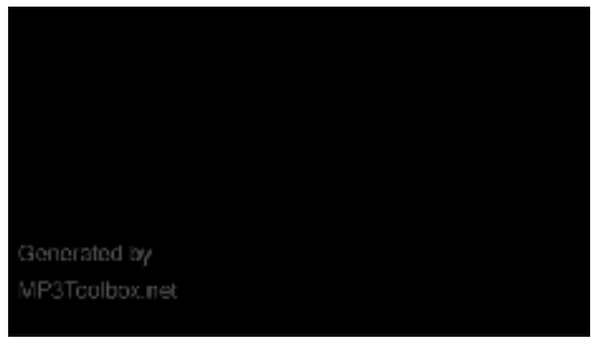
Yamins and DiCarlo, Nature Neuroscience (2016)

But, most tasks have a temporal component…

Temporal processing is crucial for our perception of sounds:



Speech 1



Speech 2

What's the difference between these two sounds?

But, most tasks have a temporal component…

Temporal processing is crucial for our perception of sounds:

Speech 1

Speech 2

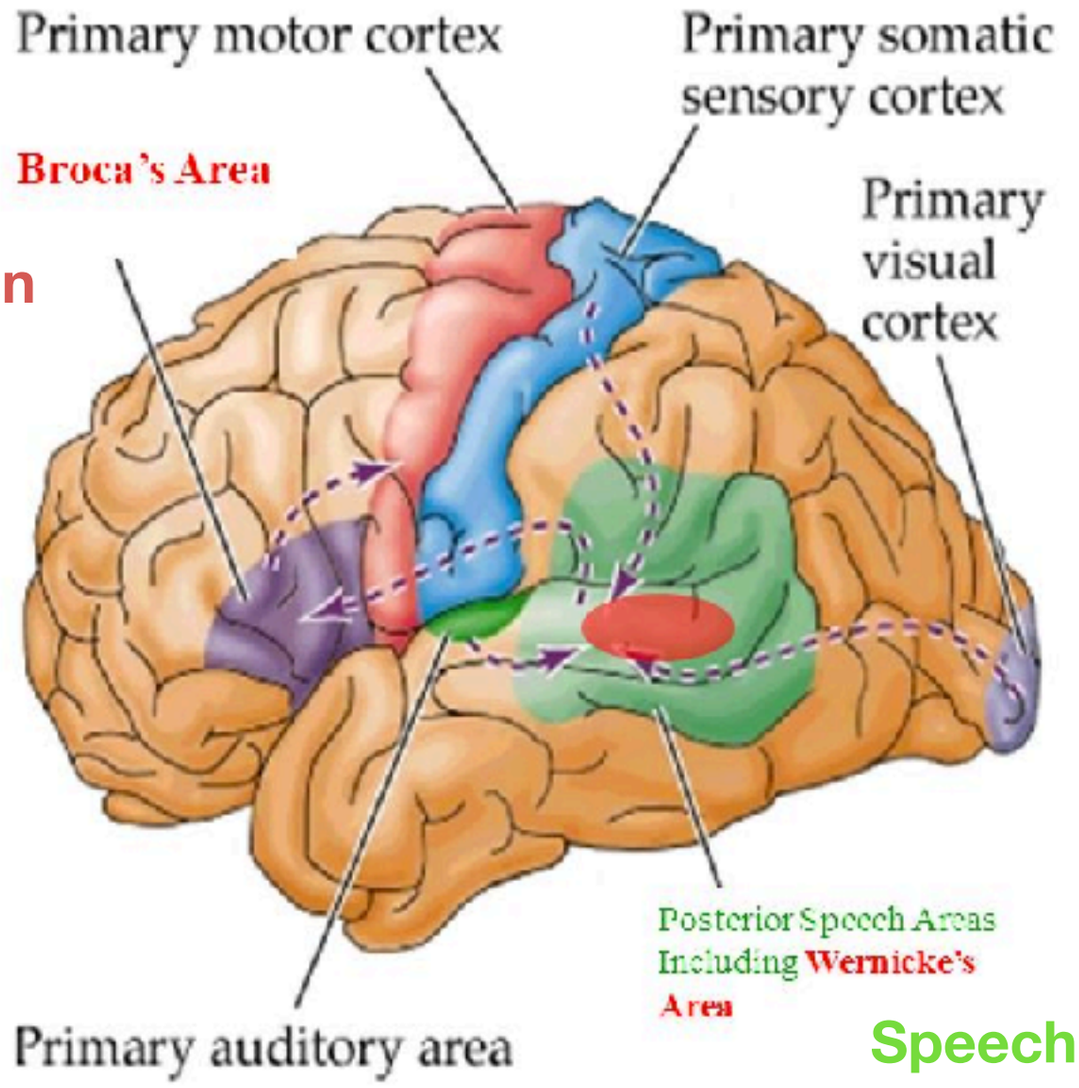
What's the difference between these two sounds?
They have the same content,
but the first is the reserved of the second.

Outline

1. **Temporal processing:** language processing and auditory cortex
2. **Recurrent neural networks**
3. **Classical networks:** Hopfield networks and Boltzmann machines
4. **Reservoir computing:** Echo networks and liquid state machines

Temporal processing

Language in the brain



Speech production

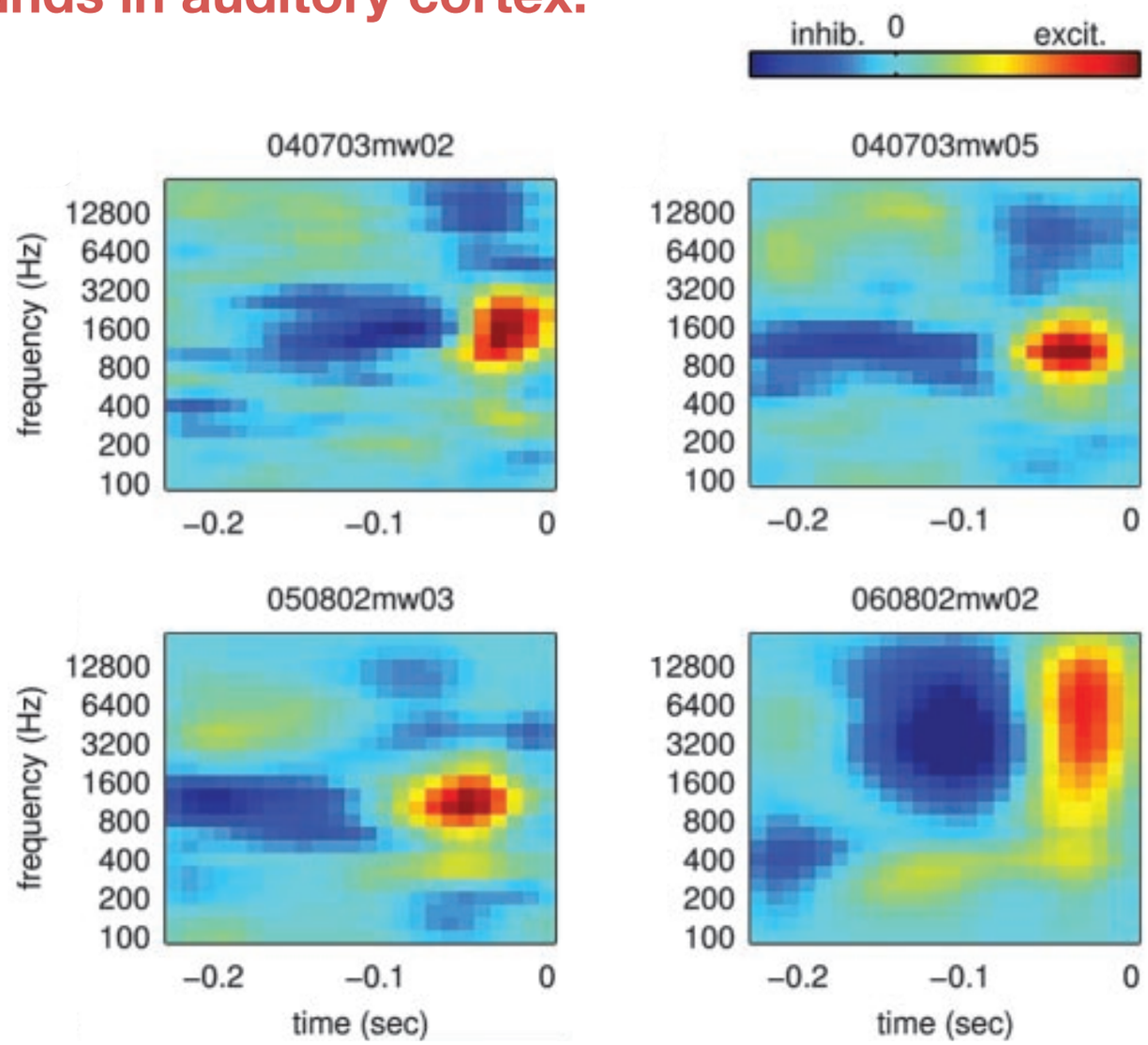
Speech comprehension

Temporal processing

Receptive fields in auditory cortex

Receptive fields for natural sounds in auditory cortex:

Note that these receptive fields are also localised in time and frequency domain, similar to receptive fields observed in the visual cortex.

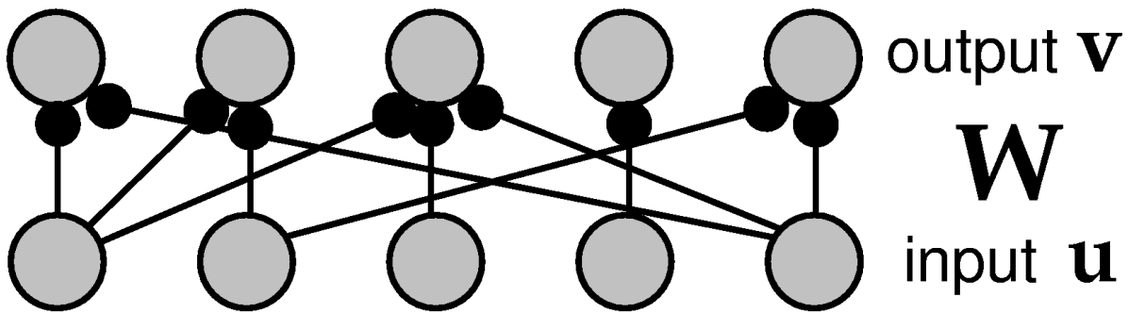


Machens et al. JNeurosci (2004)

How to model temporal processing?

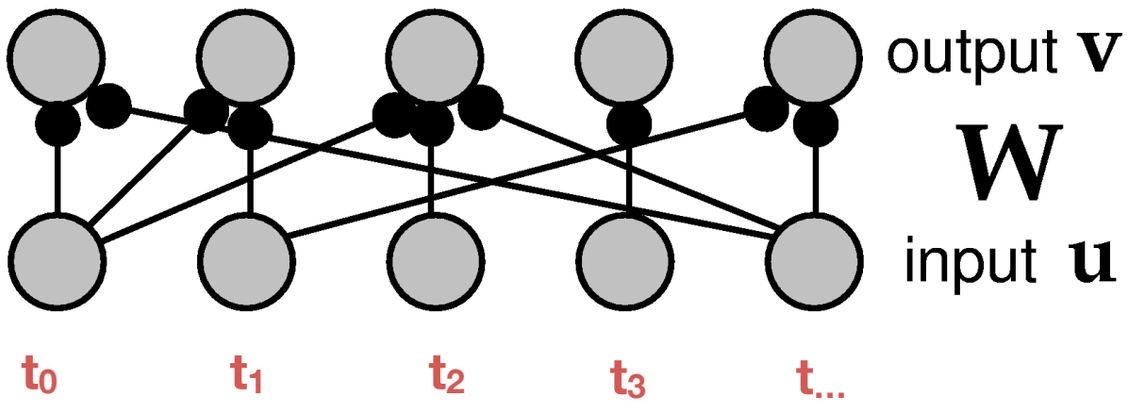
Feedforward networks do not model time

Feedforward networks do not model time explicitly.



Feedforward networks do not model time

One option is to consider time as different inputs:

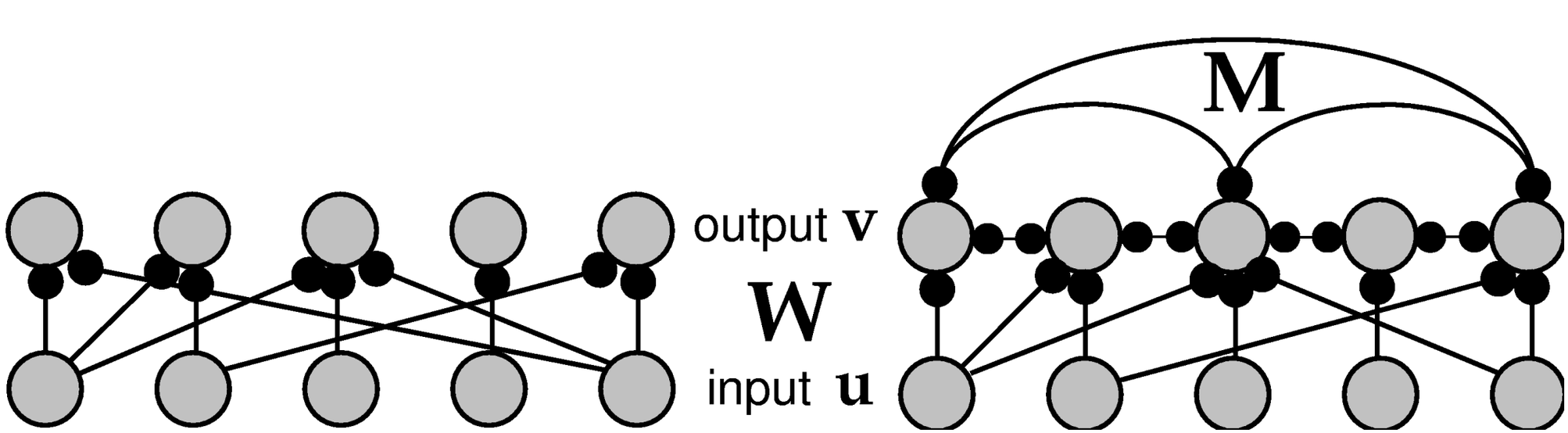


Feedforward vs recurrent networks

But it is more adequate to use *recurrent neural networks* to model time explicitly.

Feedforward

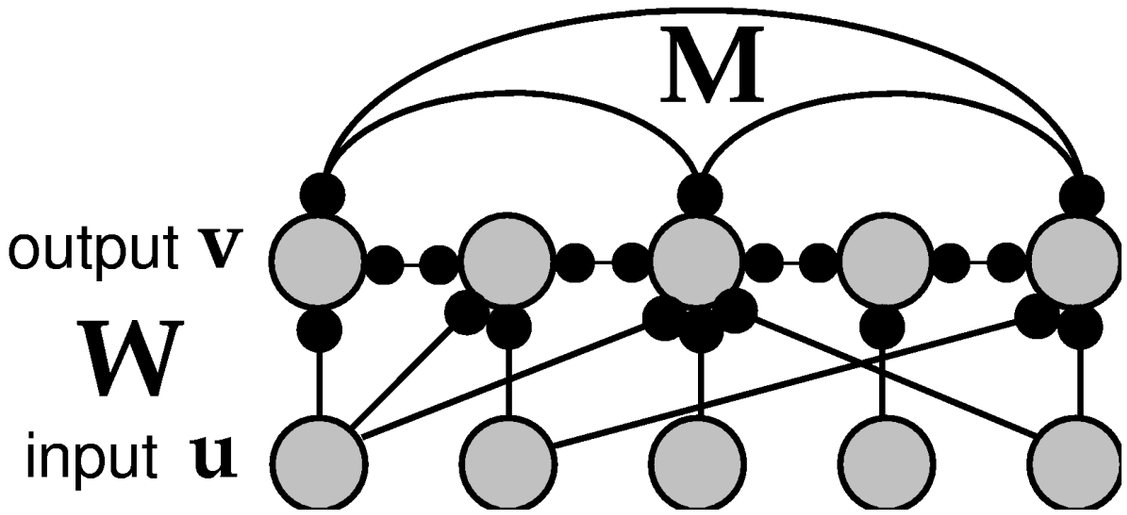
Recurrent



Dayan and Abbott book (2001)

Recurrent neural networks

mathematical models



Discrete-time:

[often used in machine learning]

$$v_{t+1} = rv_t + f(Wu_t + Mv_t)$$

r, is the activity decay

Continuous-time:

[often used in computational neuroscience]

$$\tau_r \frac{dv}{dt} = -v + f(Wu + Mv)$$

τ_r, is timeconstant of activity decay

Dayan and Abbott book (2001)

Application of RNNs in neuroscience:
How does the brain remember eye position?

How does the brain remember where it is looking at
despite **constant blinking**? Using some form of memory?

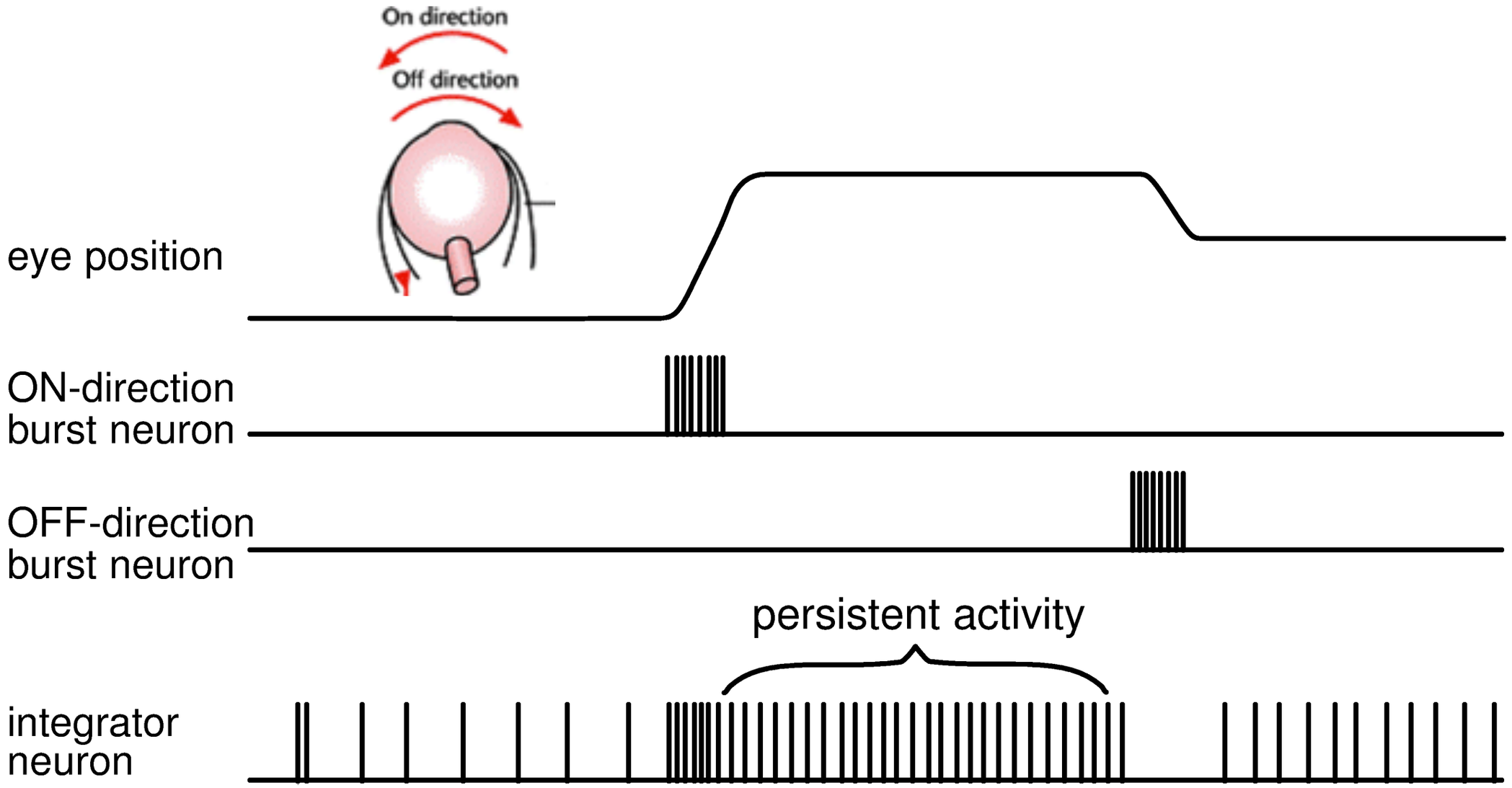
Eye blinking:

Seung et al, (2000)

How does the brain remember eye position?

Neural integrator for the oculomotor system

How does the brain remember where it is looking at?



Seung et al, (2000)

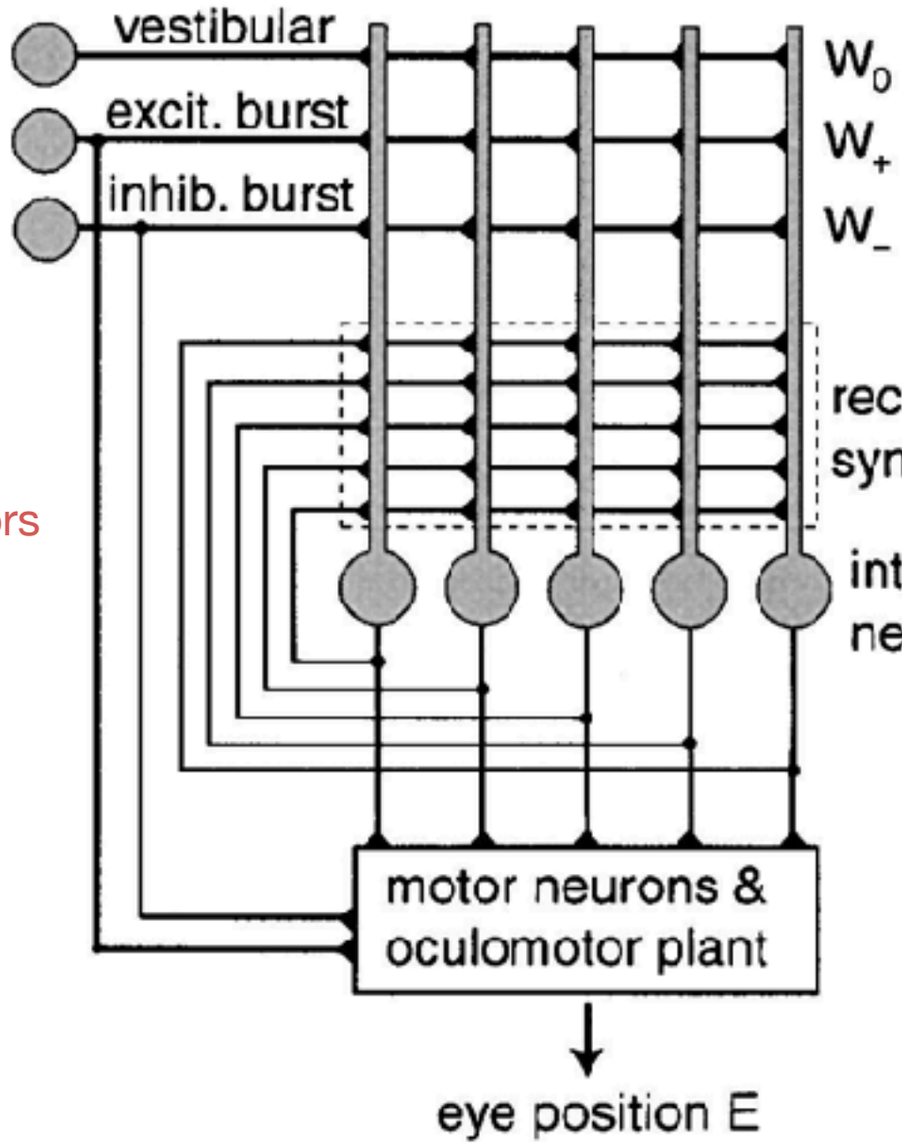
How does the brain remember eye position?

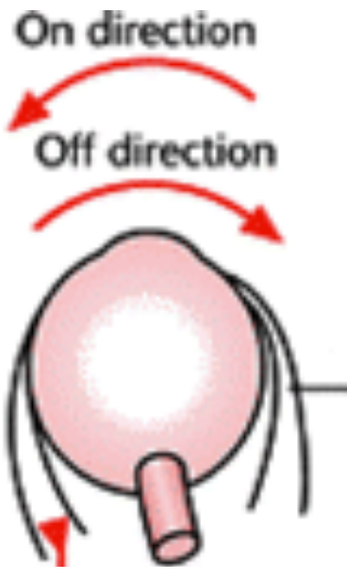
Neural integrator for the oculomotor system

body/head position input:

detects **on-direction** change:

detects **off-direction** change:





neural integrators (memory):

Seung et al, (2000)

Group discussion
groups of 2-3 (5min)

$$v_{t+1} = rv_t + f(Wu_t + Mv_t)$$

r defines the decay of activity in a given neuron,

what should it be to have a **perfect neural integrator**?

and what should it be for different **tasks**?

For simplicity, we assume here that diag(M) = 0, so no self recurrence

Group discussion
groups of 2-3 (5min)

r defines the decay of activity in a given neuron,
what should it be to have a **perfect neural integrator**?
and what should it be for different **tasks**?

For ***r=1***, you have a *perfect integrator*, the neuron will 'remember' exactly previous inputs
For ***r<1***, the activity decays back to baseline
For ***r>1***, activity grows exponentially over time (i.e. *runaway activity*)
So you would choose/optimise ***r*** depending on the timescale of memory you need for a given task.

$$v_{t+1} = rv_t + f(Wu_t + Mv_t)$$

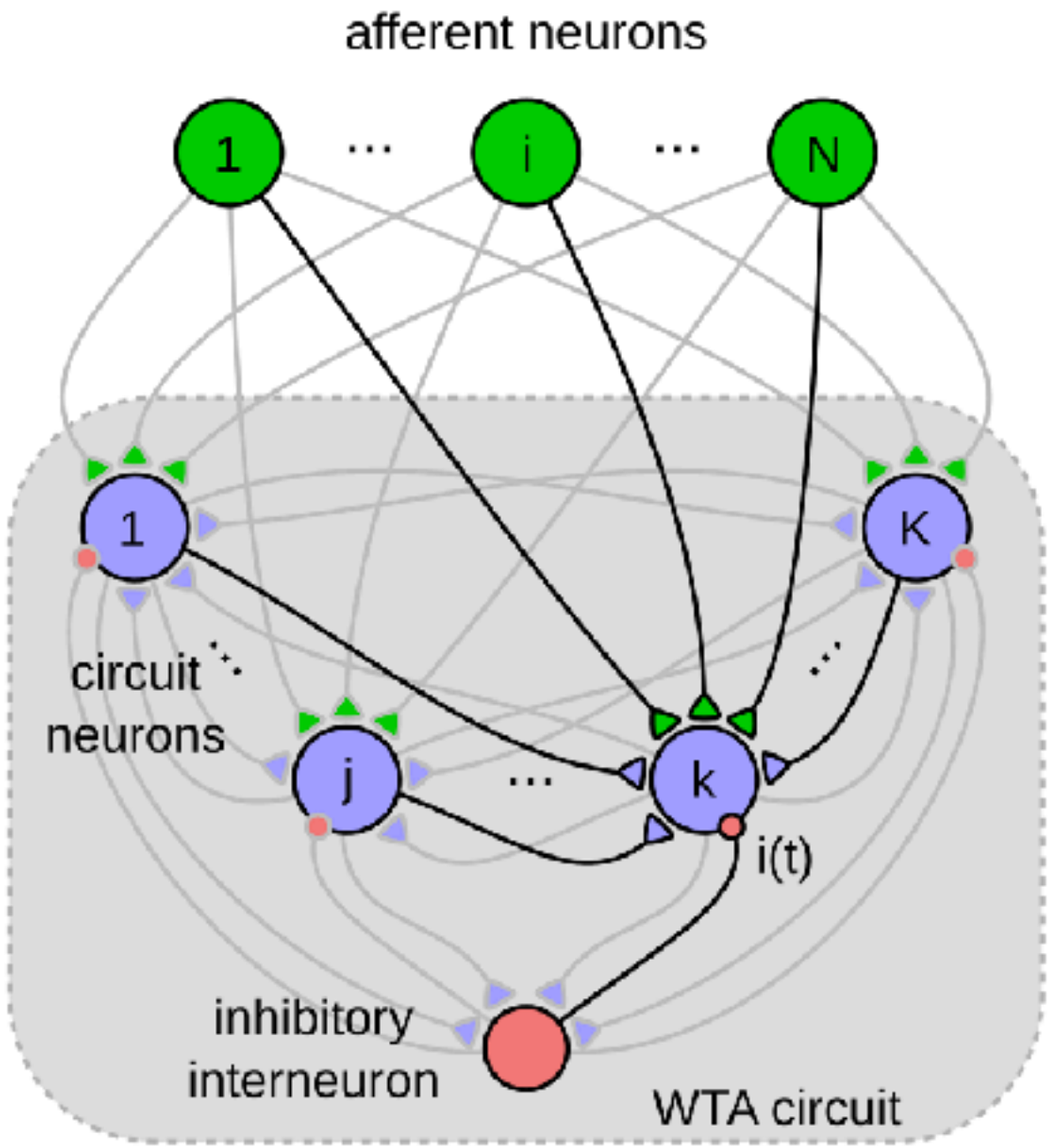
For simplicity, we assume here that diag(M) = 0, so no self recurrence

Selecting only the strongest response

Winner-take-all (WTA) network

WTA operation is similar to applying the *max* operator to the response.

Implementation of a **WTA circuit** via recurrent inhibitory neurons:

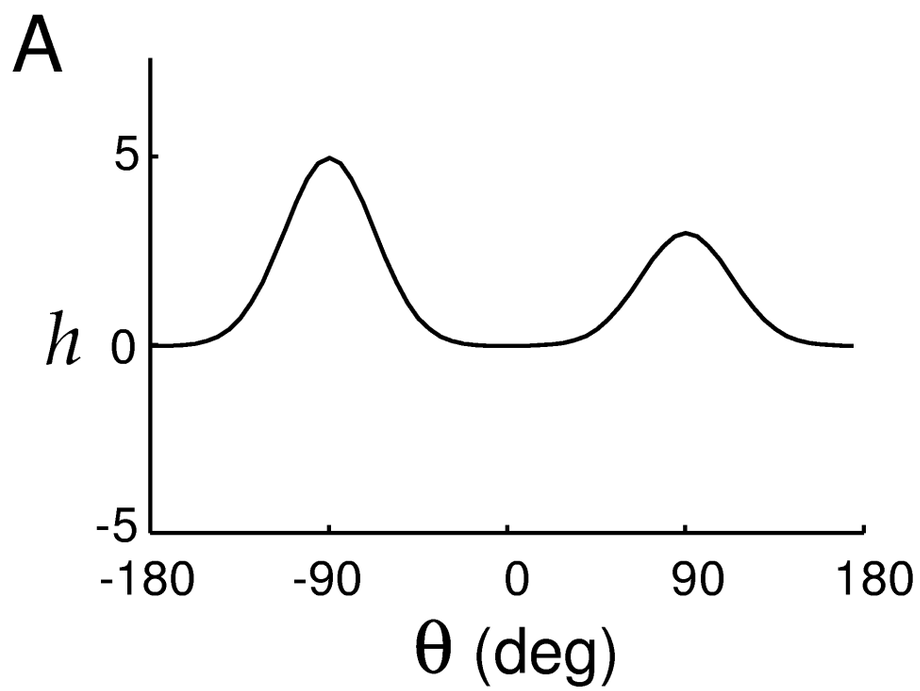


Kappel et al. PLoS CB (2014)

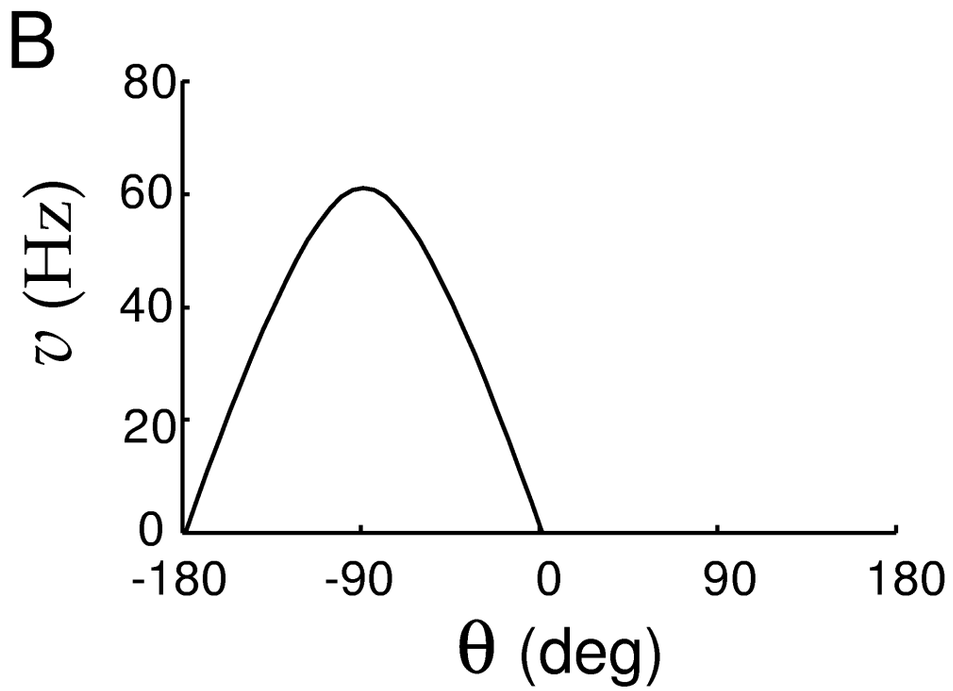
Selecting only the strongest response

Winner-take-all network

Before winner-take-all



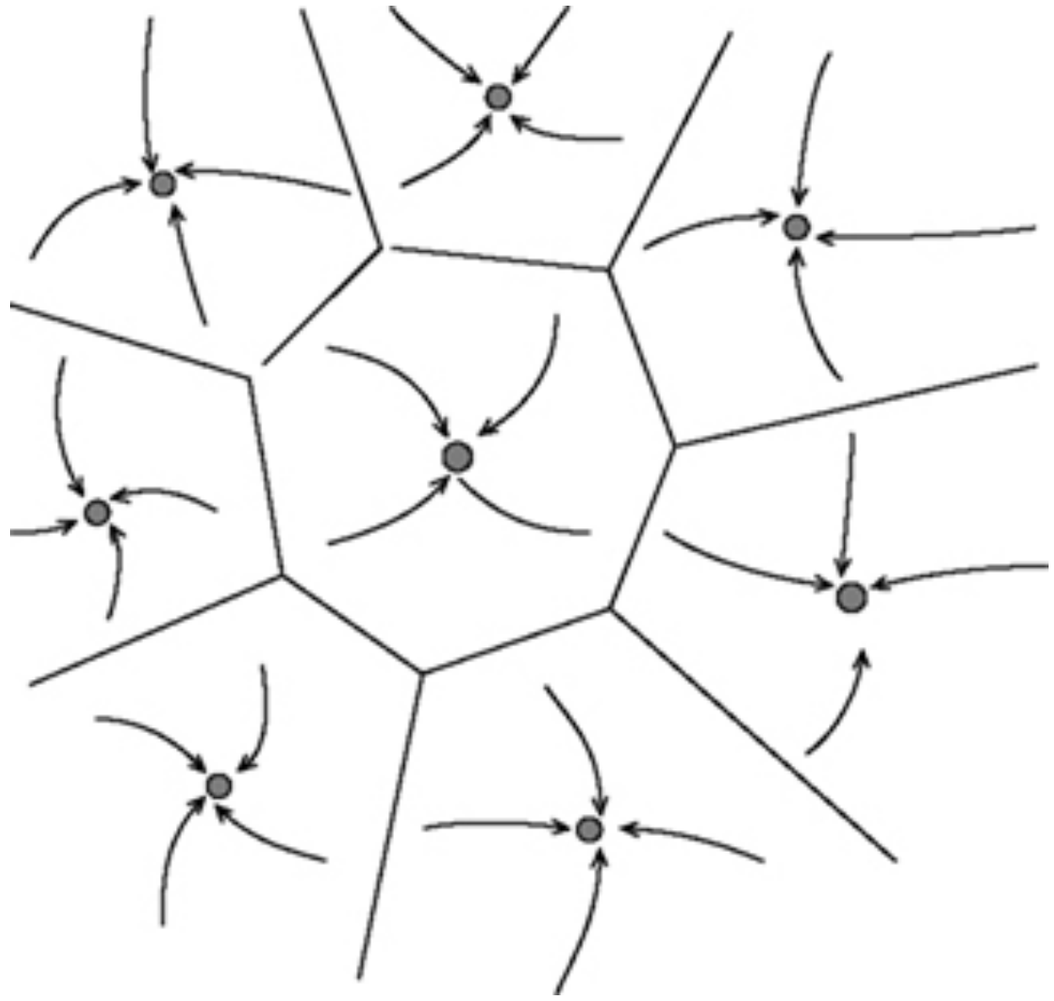
After winner-take-all



Dayan and Abbott book (2001)

Attractor neural networks

Attractor neural networks are dynamic systems that contain a certain number of attractor states, with low energy to which the network dynamics converge.

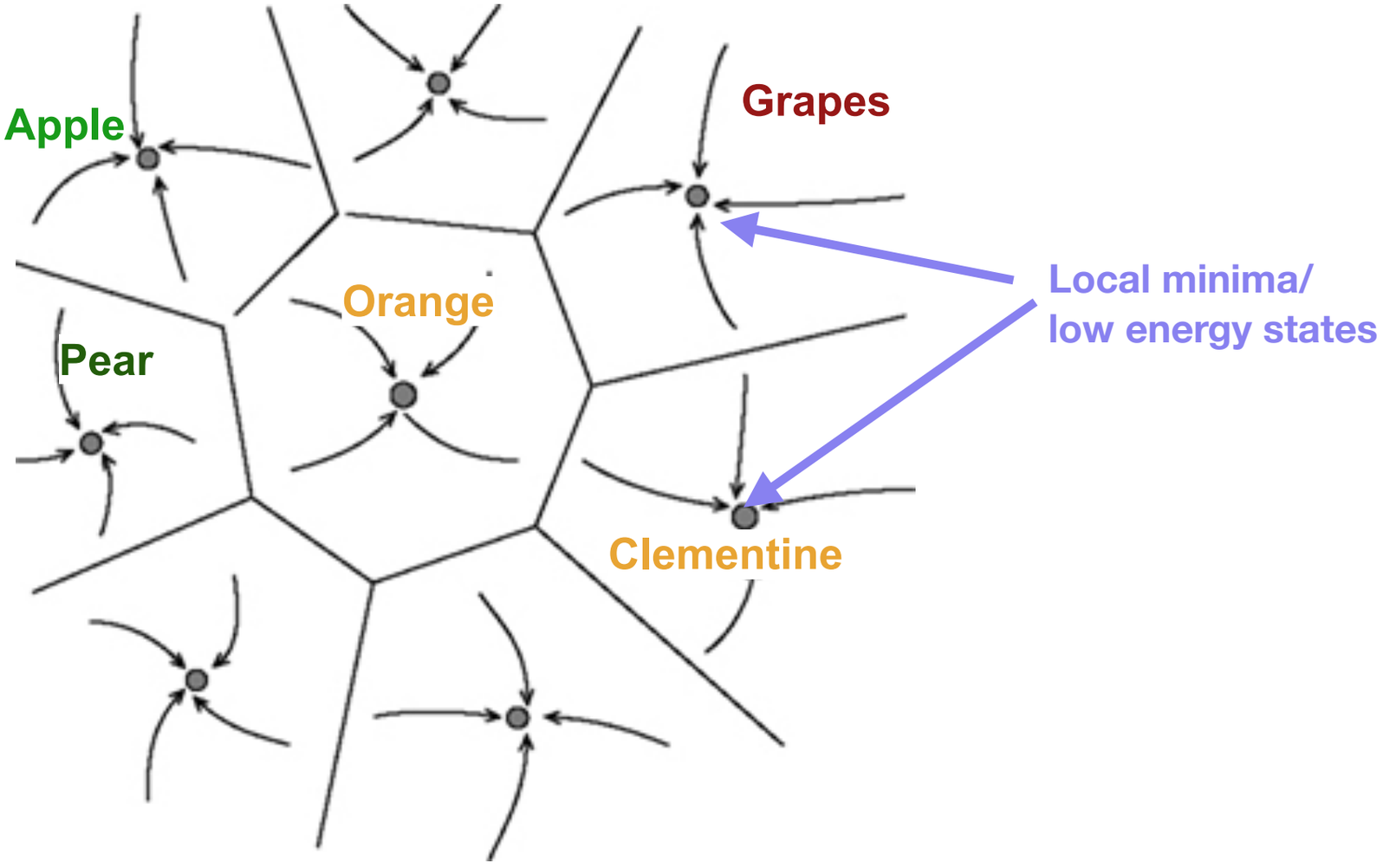


Dayan and Abbott book (2001)

Attractor neural networks

Hopfield networks

Hopfield networks are content-addressable ("associative") memory networks with binary threshold neurons. The network is guaranteed to converge to a local minima, attractor/local minima states represent different memories.



Representation of energy landscape

Hopfield 1982

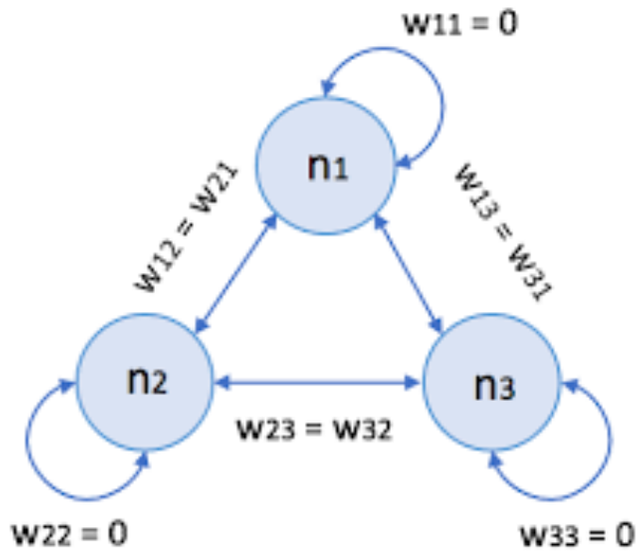
Attractor neural networks

Hopfield networks

Two key properties:

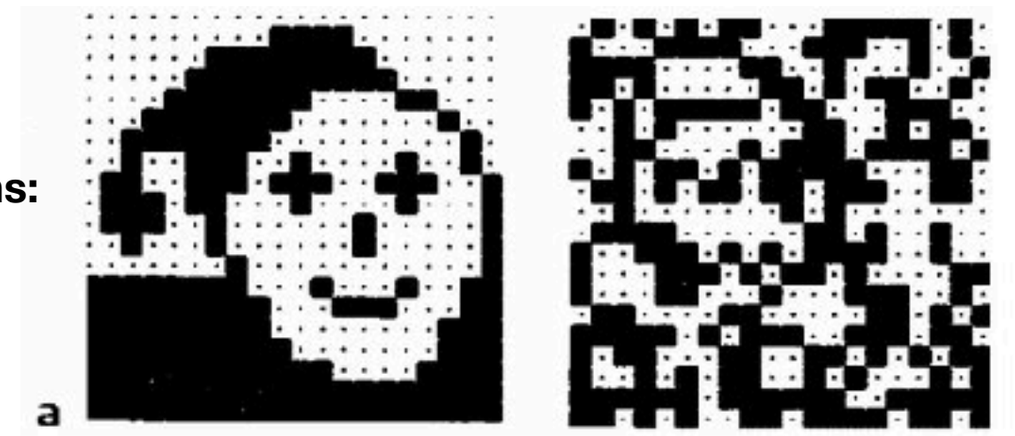
$m_{ii} = 0$ no unit has a connection with itself

$m_{ij} = m_{ij}$ connections are symmetric

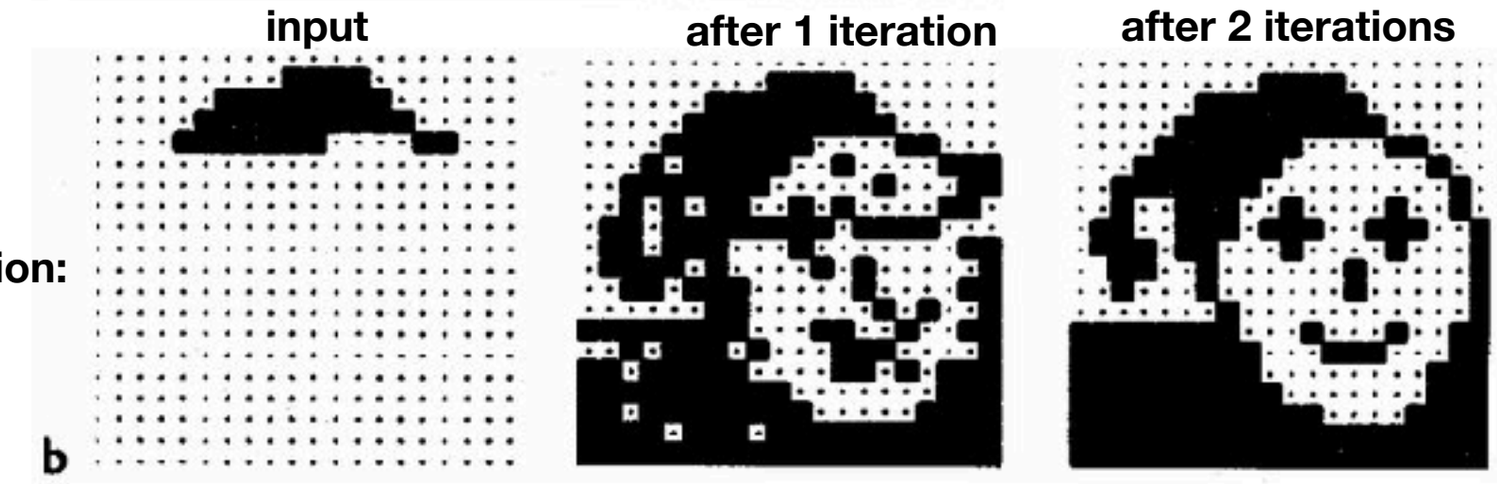


Hopfield connectivity, *W* here is the same as *M* in our lectures

Trained patterns:



Example of pattern completion:



Hopfield 1982

Attractor neural networks

Boltzmann machines

Boltzmann machines are the stochastic version of Hopfield networks.

Total input:

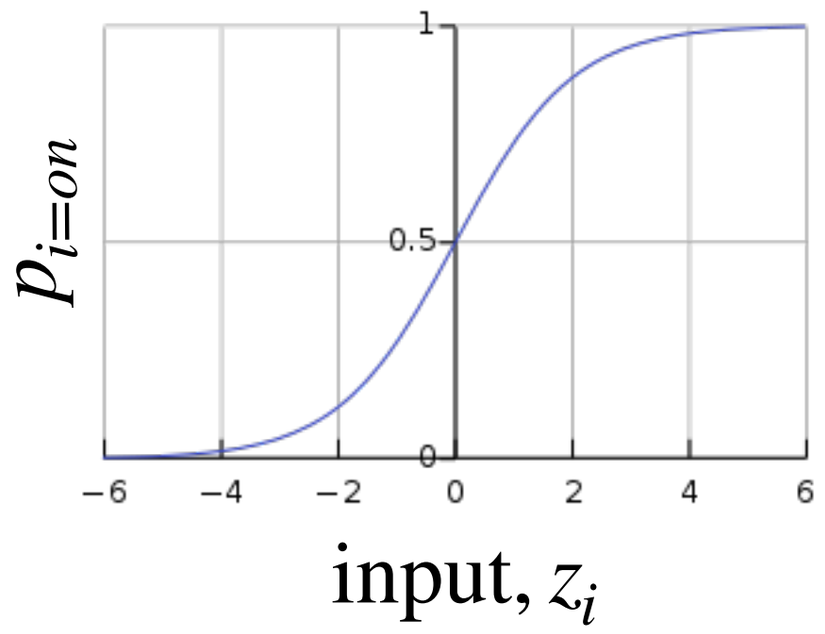
$$z_i = b_i + \sum_j u_j w_{ij} + \sum_x v_x m_{ix}$$

external input

recurrent input

Probability of turning on a given neuron *i*:

$$p_{i=on} = \frac{1}{1 + \exp(-\frac{z_i}{T})}$$



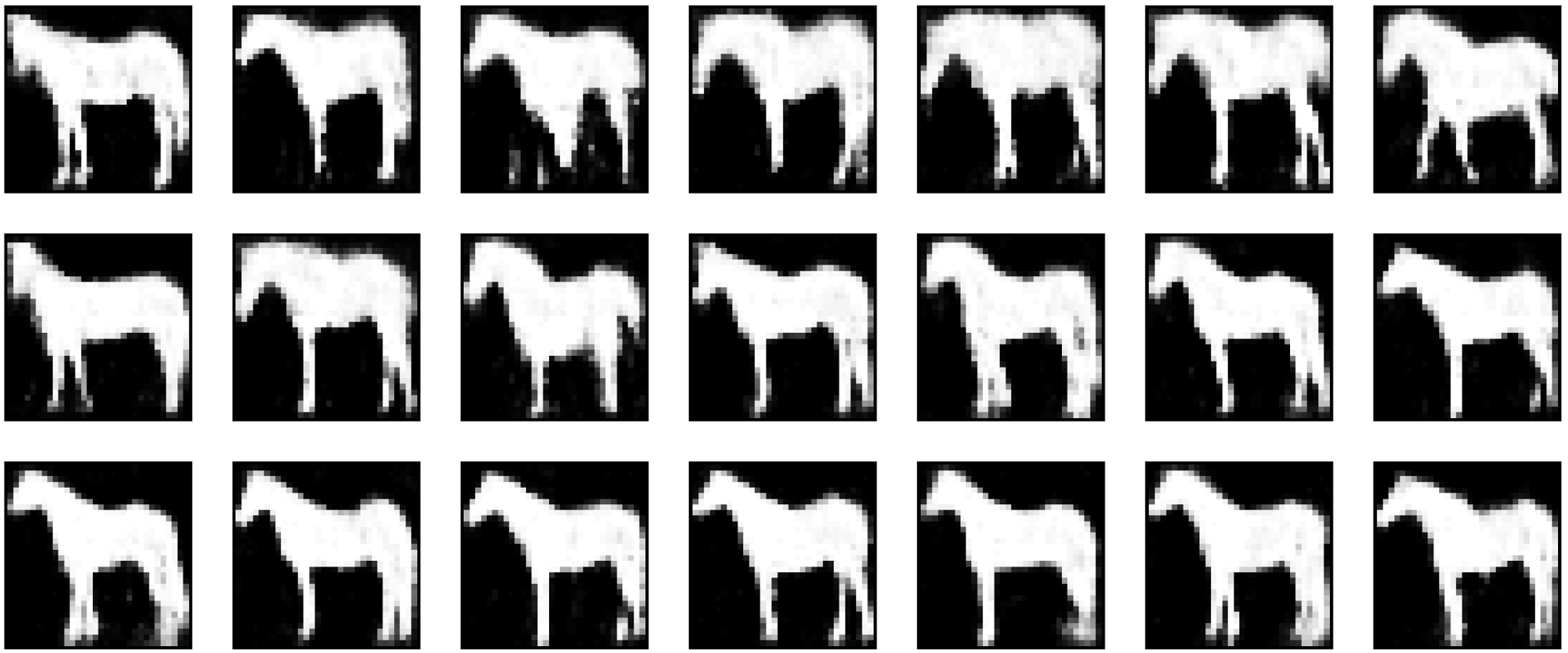
For temperature, T=0 we obtain Hopfield networks.

www.scholarpedia.org/article/Boltzmann_machine

Autoassociative neural networks

Boltzmann machines

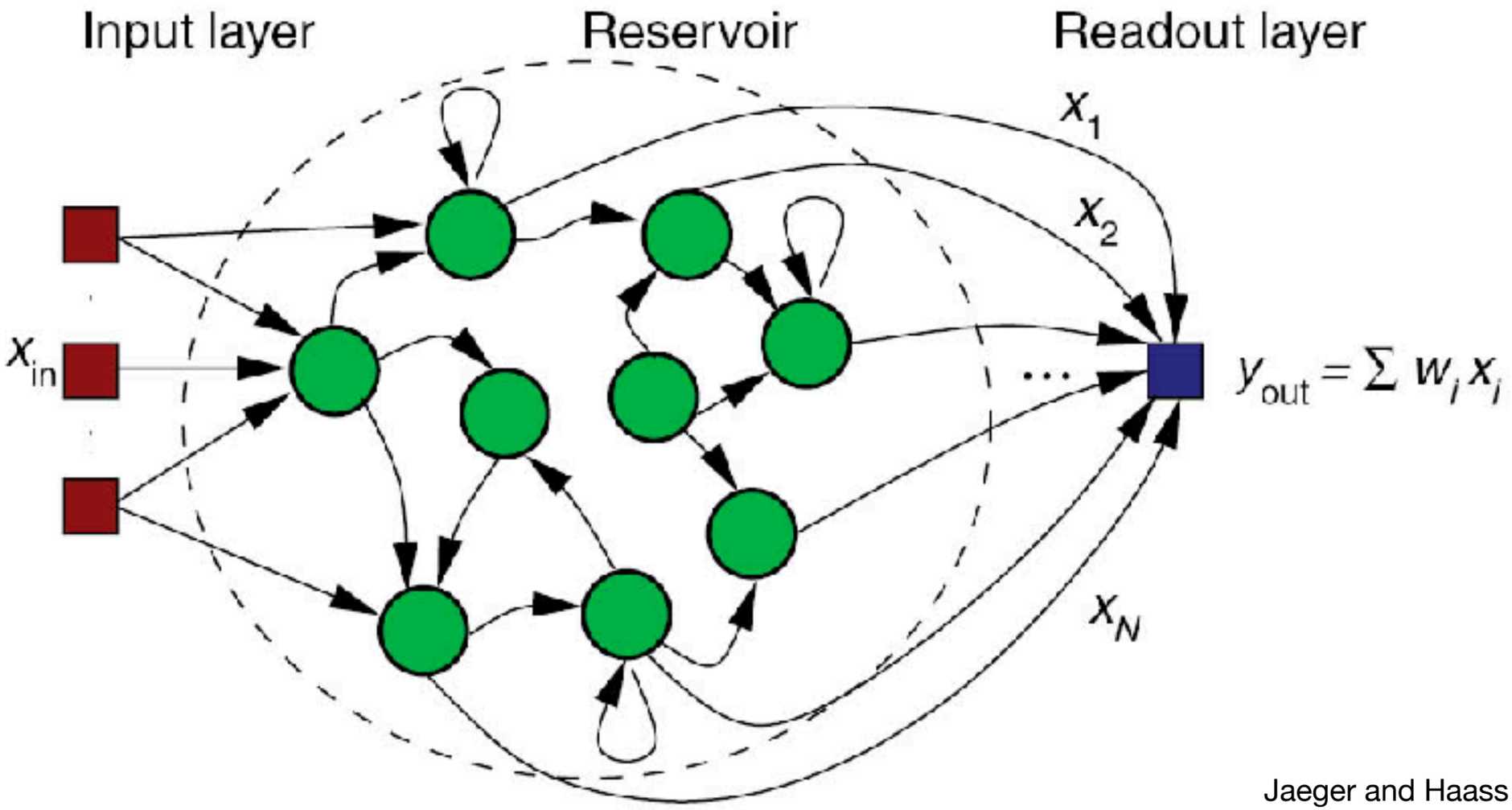
Boltzmann machines are good generative models. Samples from a Boltzmann machine after it learned the shape of horses:



Eslami et al. (2014)

Reservoir computing

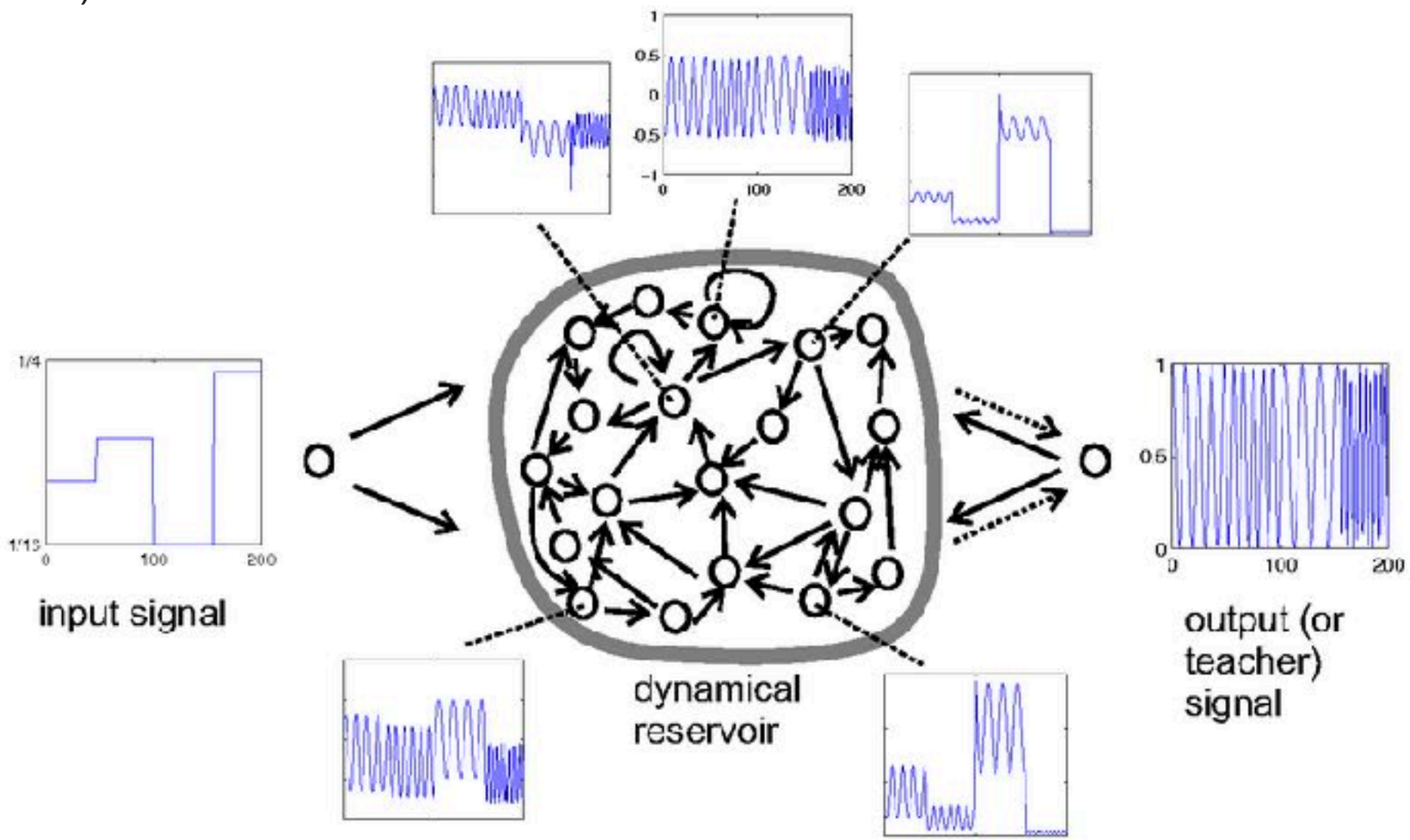
Key idea: Use a large pool of (sparsely connected) neurons which generate a large repertoire of dynamics (the reservoir with **fading memory**), and learn a simple read-out to combine the interesting dynamics into an useful output. It expands the input into higher dimensions to be easier to separate by the output (**input separability**).



Jaeger and Haass 2004

Reservoir computing: echo state networks

Echo state networks: is the most common rate-based (i.e. neurons are modelled at the level of firing rates) reservoir network.

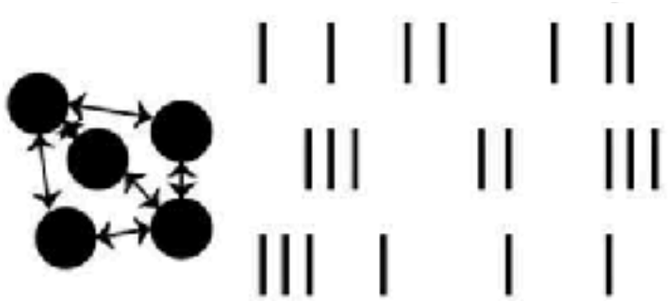


Jaeger and Haass 2004

Reservoir computing: liquid state machines

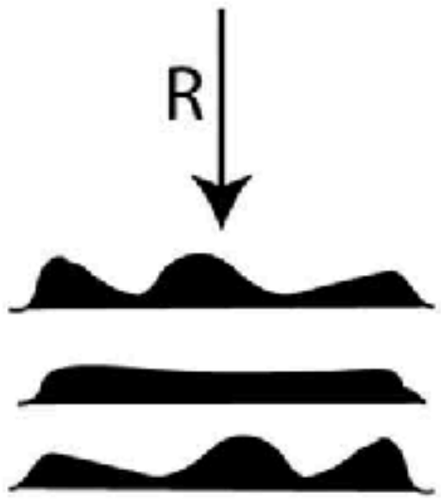
Liquid state machines: similar to echo state networks, but neurons are modelled at the level of individual spikes, a more realistic model of real neurons.

Liquid state machines



Spiking model
(model individual spikes at specific points in time)

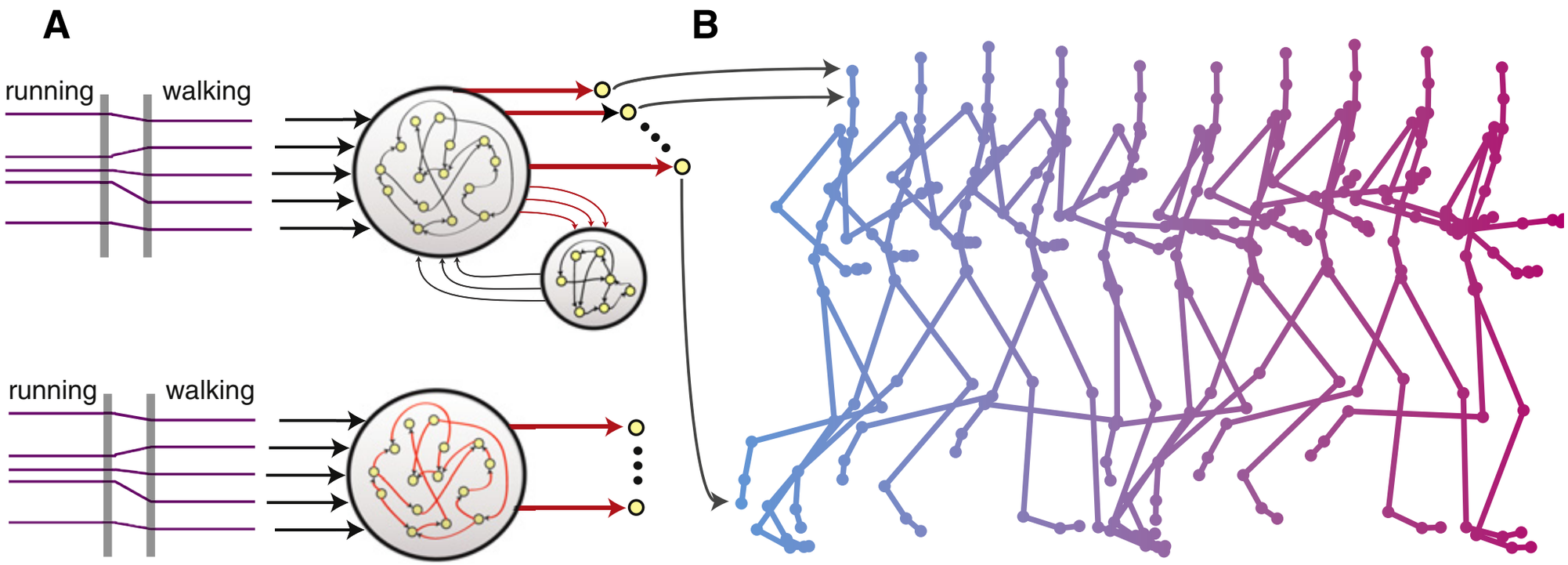
Echo state networks



Rate model
(only captures the rate of spiking)

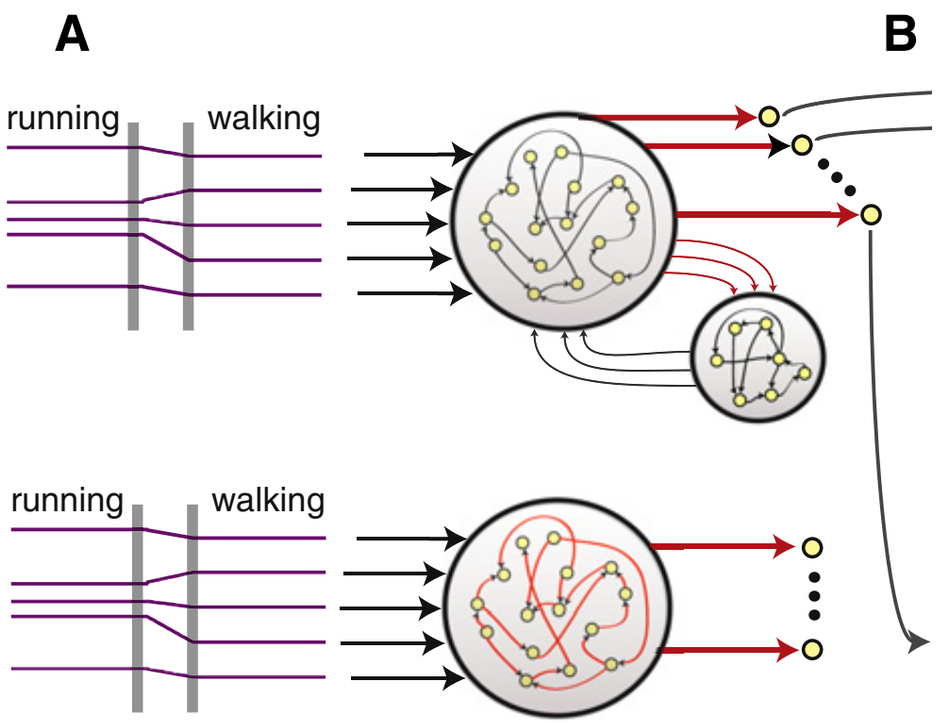
Maass et al. Neural Computation (2002)

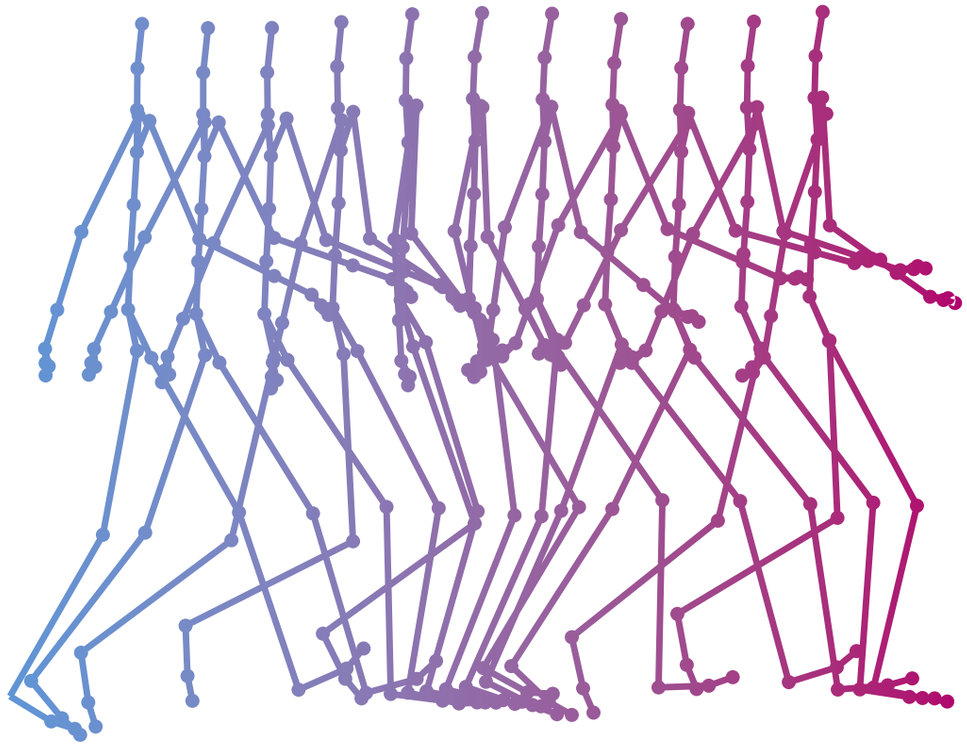
Reservoir computing:
a network that can learn to **run**/walk



Sussillo and Abbott Neuron (2009)

Reservoir computing:
a network that can learn to run/**walk**

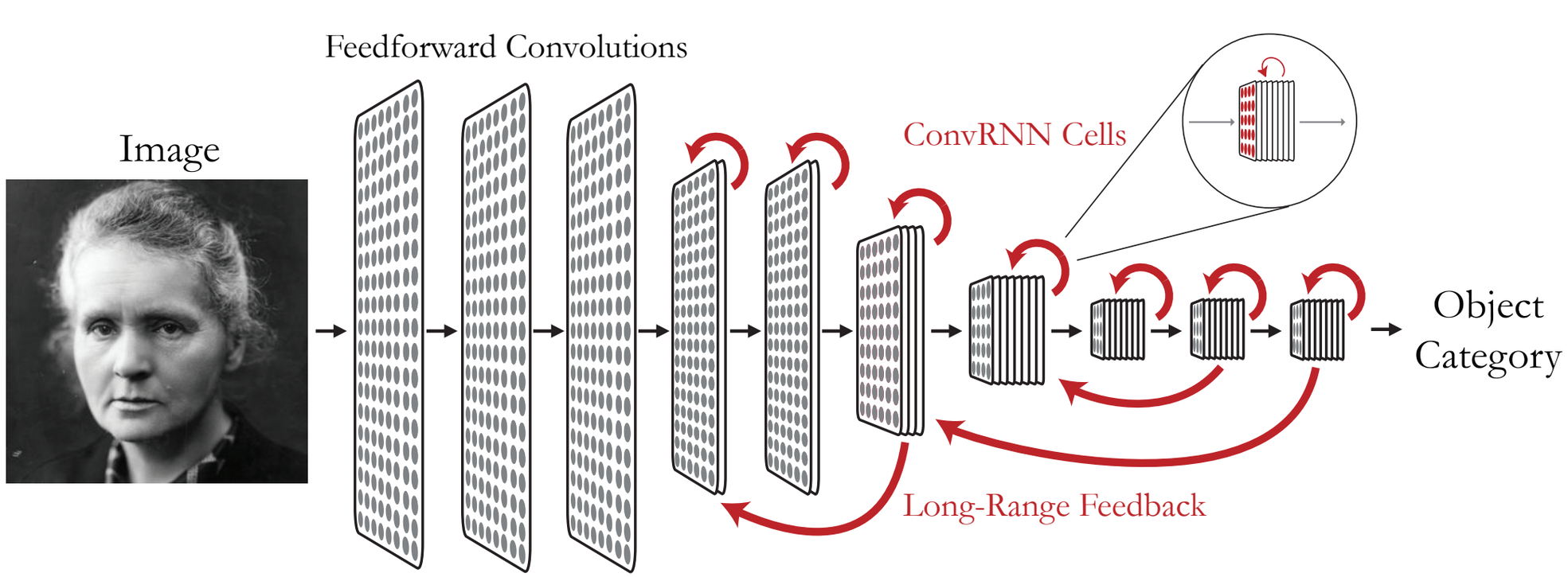




Sussillo and Abbott Neuron (2009)

Making CNNs recurrent

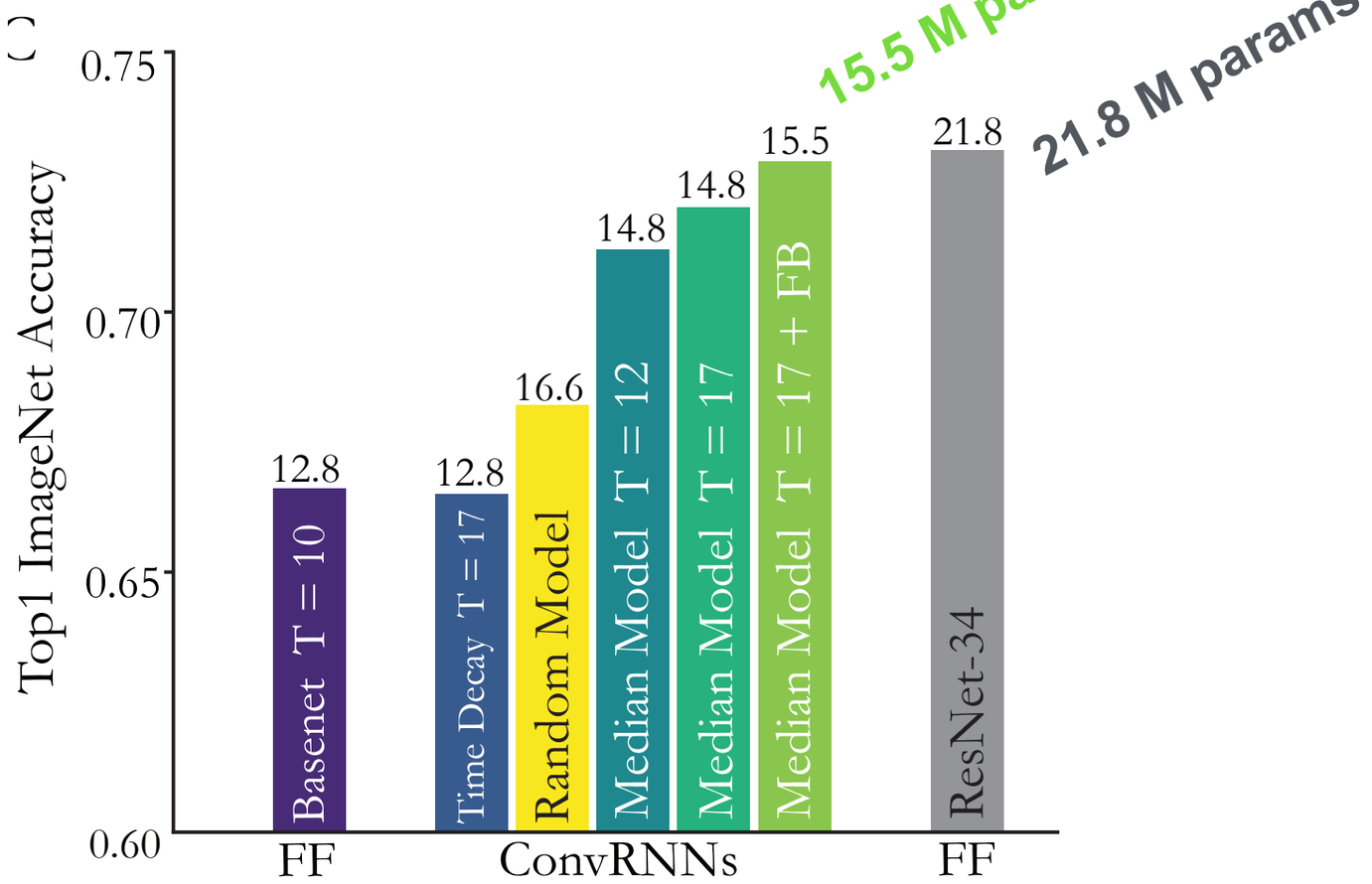
Most brain areas exhibit a high degree of recurrence (both within and between brain areas), recent developments show that if convolutional neural networks (CNNs) are extended with specific forms of recurrence they can outperform purely feedforward CNNs.



Nayebi et al. NeurIPS (2018)

Making CNNs recurrent

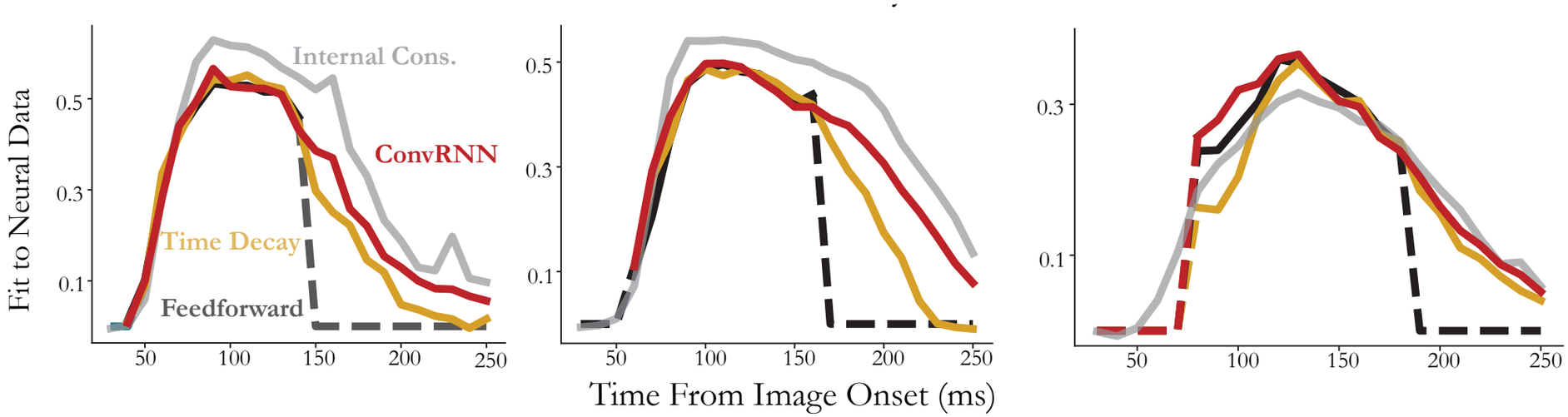
Nayebi et al. searched over multiple forms of recurrent CNNs (ConvRNNs) and found that they can still obtain good performance, while using *less depth* as current deep networks, and *less parameters*.



Nayebi et al. NeurIPS (2018)

Making CNNs recurrent

Nayebi et al. also found that **ConvRNNs** can capture the temporal responses in the cortex, better than standard feedforward deep networks.



Nayebi et al. NeurIPS (2018)

Summary

1. **The brain relies on temporal processing**

2. **Recurrent neural networks are more readily applicable to temporal processing**

3. **Associative attractor networks can store memories as low energy states**

4. **Reservoir computing explores dynamical properties of RNNs**

References

Text books:

Theoretical neuroscience: Dayan and Abbott 2001

Relevant papers:

- Seung et al, Neuro (2000)
- Hopfield, PNAS (1982)
- Jaeger and Haass, Science (2004) and Maass et al., Neural Computation (2002)

Upcoming lectures

- **L10: Neural circuits and learning: introduction**
  - Visual processing
    - **L11: Visual cortex**
    - **L11: Convolutional neural networks**
  - Learning in the brain
    - L12: Supervised learning: The backpropagation algorithm/cerebellum
    - L13: Unsupervised learning: Sparse coding and autoencoders
    - L14: Reinforcement learning: TD learning, Q learning, deep RL and dopamine
  - Temporal processing in the brain
    - L15: Auditory cortex and recurrent neural networks
    - **L16: Balanced and gated RNNs**