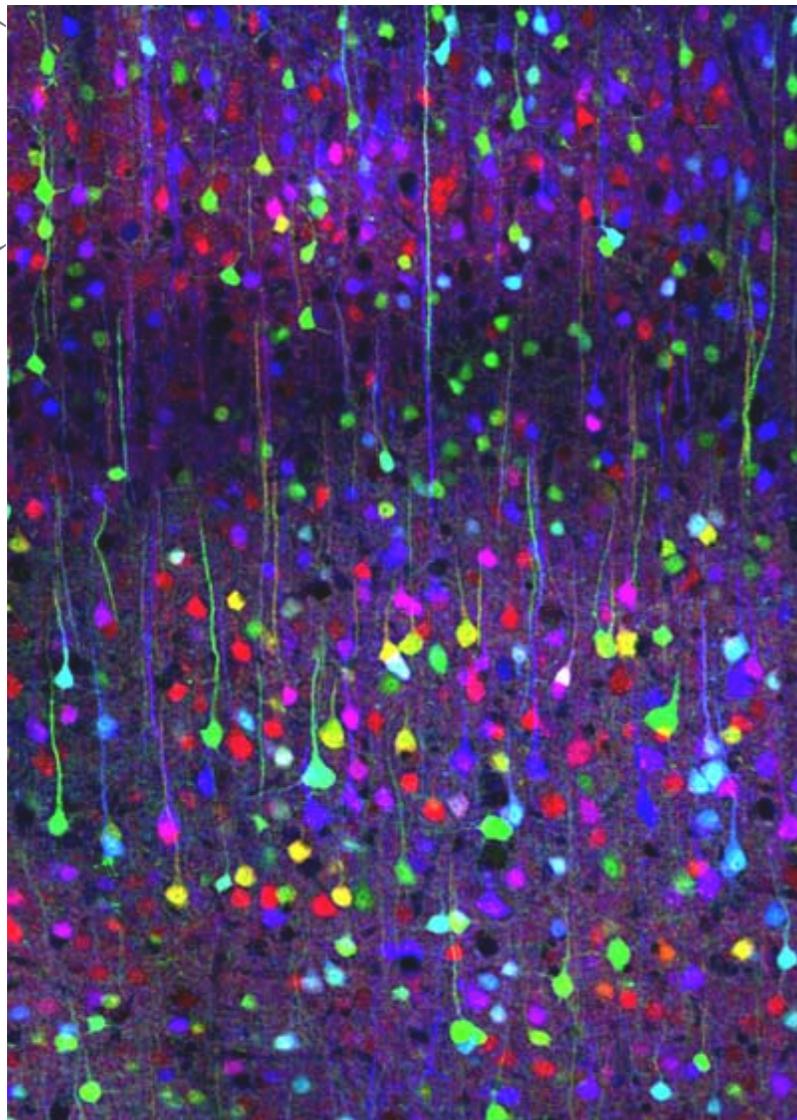


# Information Processing & the Brain 2020/2021



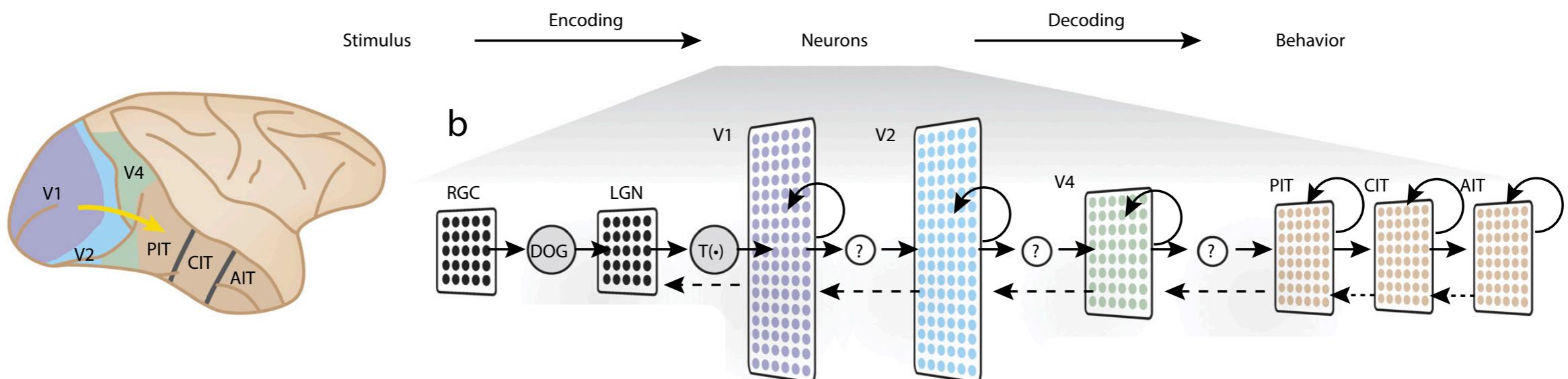
Brainbow (Litchman Lab)



## Lecture 6 Neural circuits and learning: Temporal processing and recurrent neural networks

# Previously on IPB...

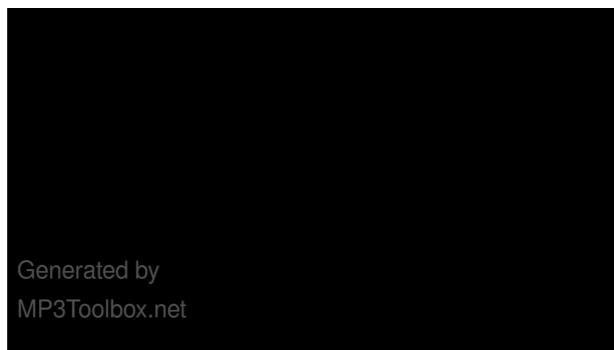
## Feedforward neural networks:



Yamins and DiCarlo, Nature Neuroscience (2016)

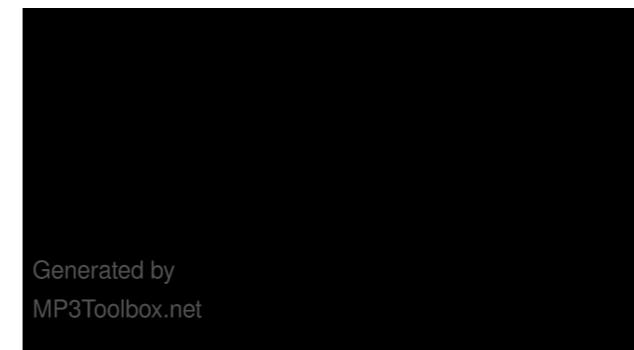
# But, most tasks have a temporal component...

Temporal processing is crucial for our perception of the world:



Generated by  
MP3Toolbox.net

Speech 1



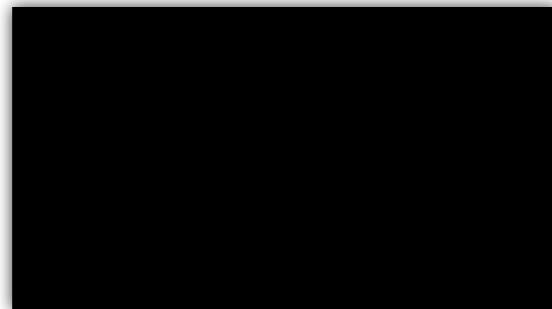
Generated by  
MP3Toolbox.net

Speech 2

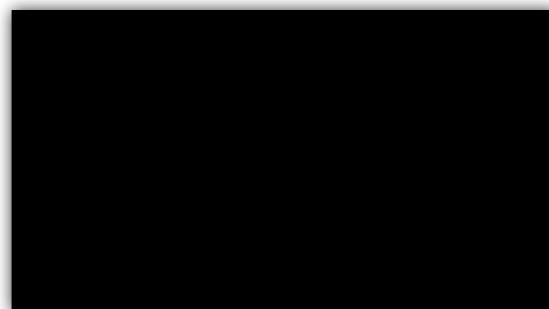
What's the difference between these two sounds?

# But, most tasks have a temporal component...

Temporal processing is crucial for our perception of the world:



Speech 1



Speech 2

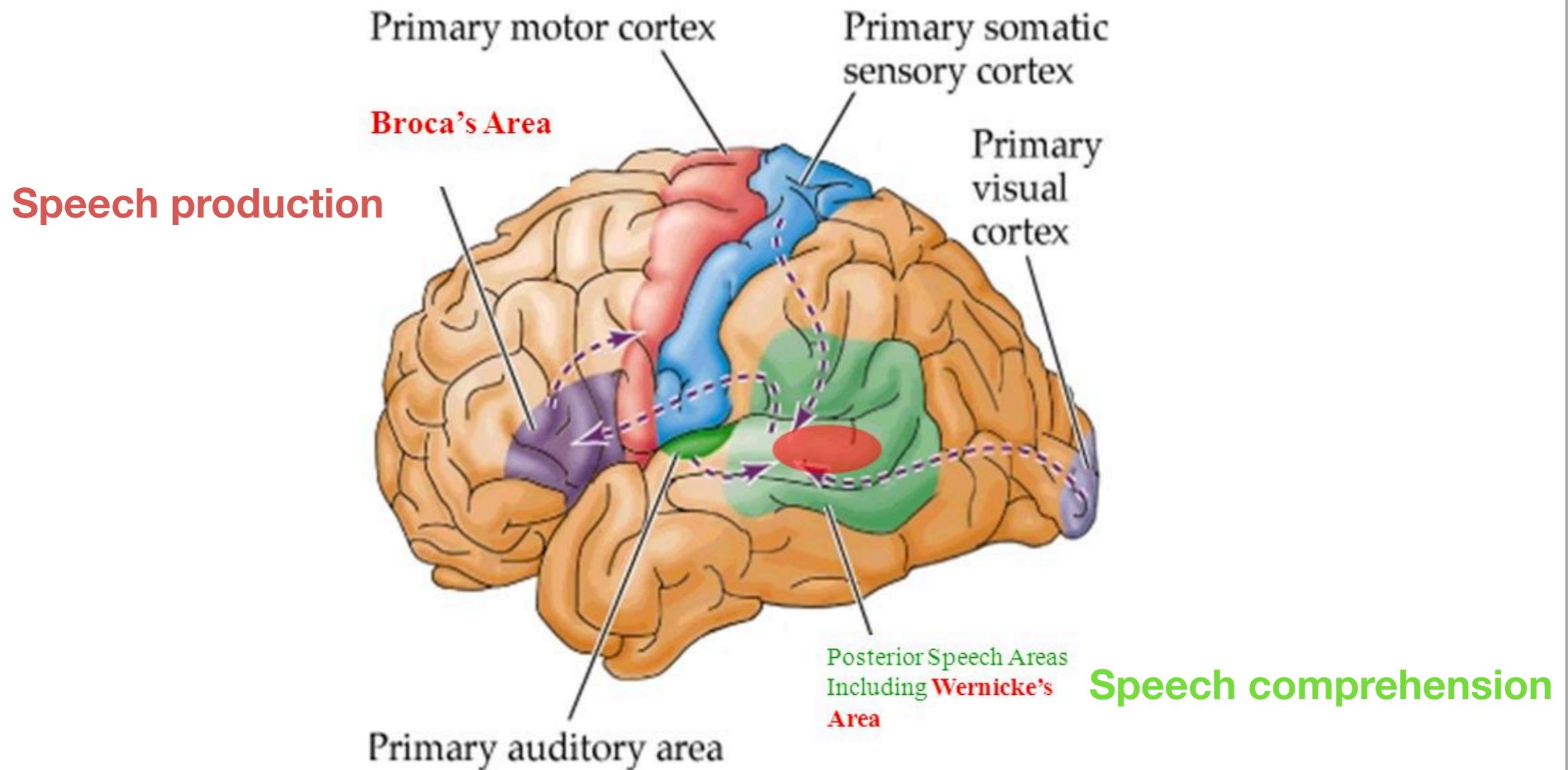
What's the difference between these two sounds?  
They have the same content,  
but the first is the reverse of the second.

# 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
- 5. Recurrent CNNs**

# Temporal processing

## Language in the brain



sound input > speech comprehension > speech prod. > motor command

auditory area > wernick's area > broca's area > motor cortex

# Temporal processing

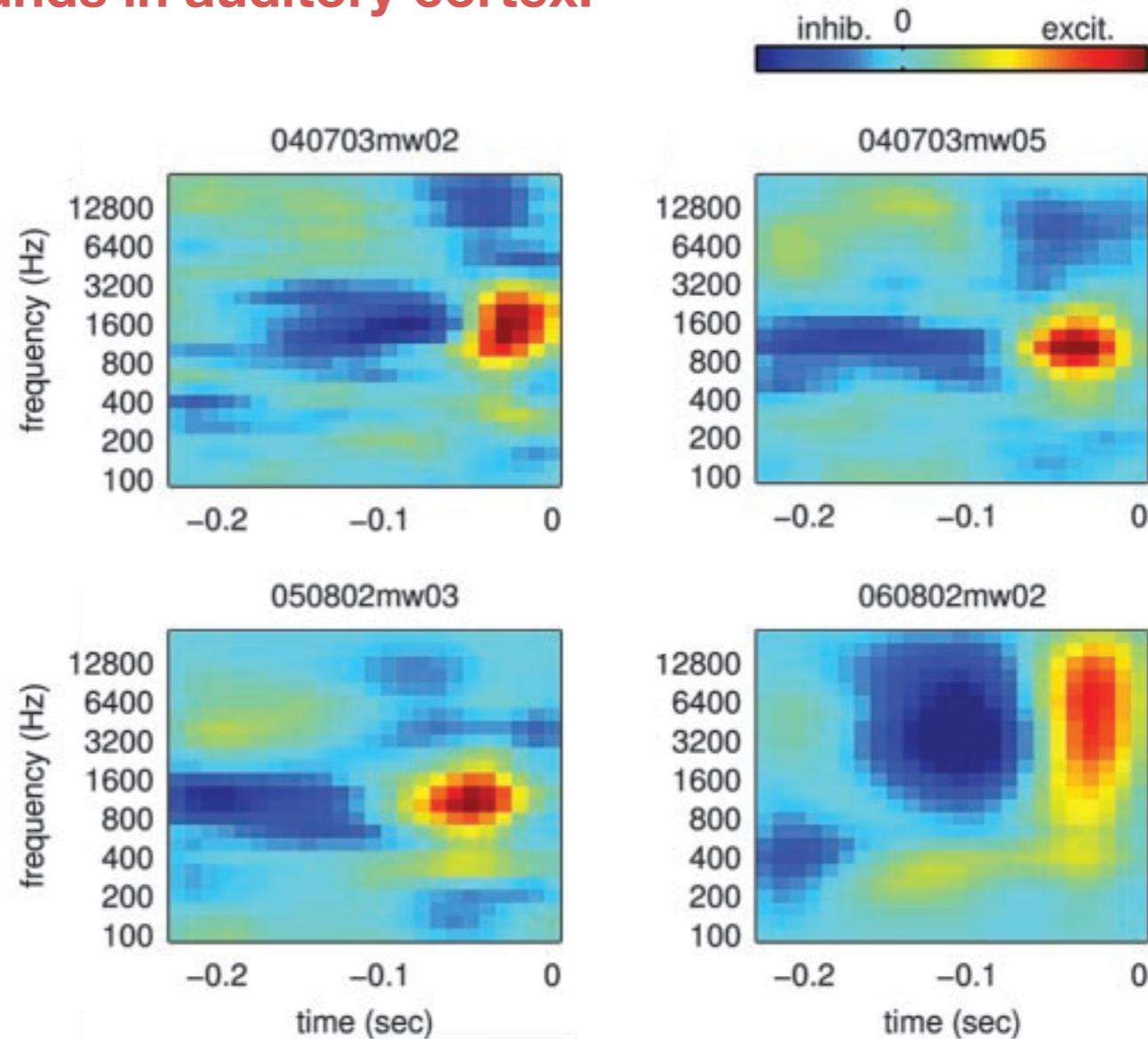
## Receptive fields in auditory cortex

### Receptive fields for natural sounds in auditory cortex:

Different panels are responses for different natural sounds.

**Red** represents increase in the neuronal activity (exc.), **blue** a decrease (inh.).

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

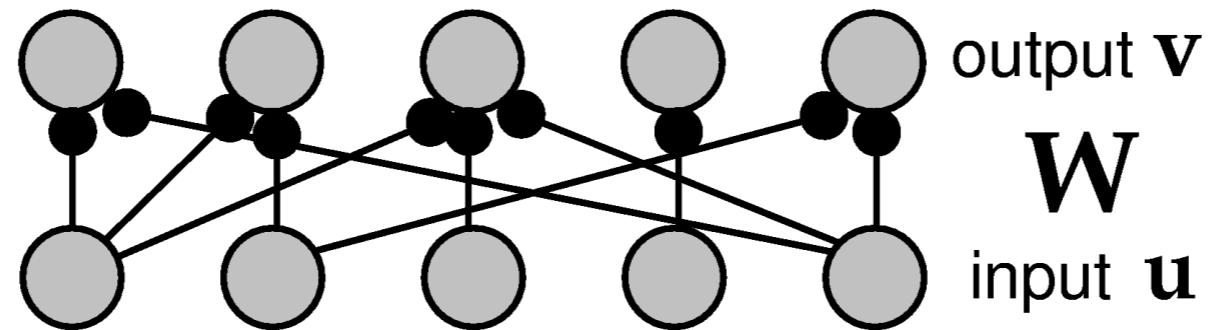


Machens et al. JNeurosci (2004)

# How to model temporal processing?

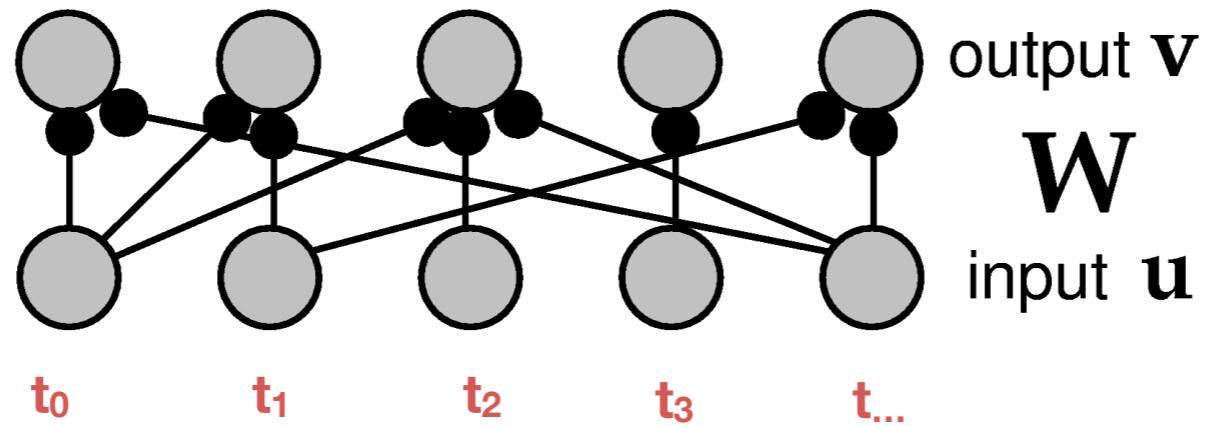
Feedforward networks do (explicitly) not model time

**Feedforward networks do not model time explicitly.**



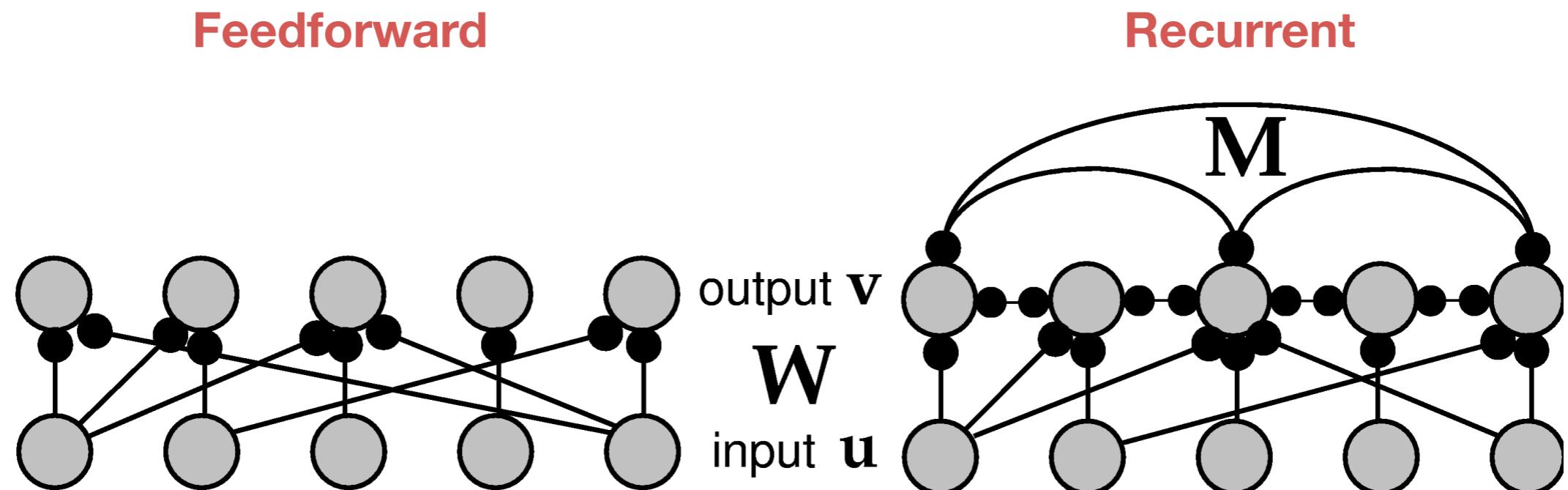
# Feedforward networks do not model time

One option is to consider time as different inputs:



# Feedforward vs recurrent networks

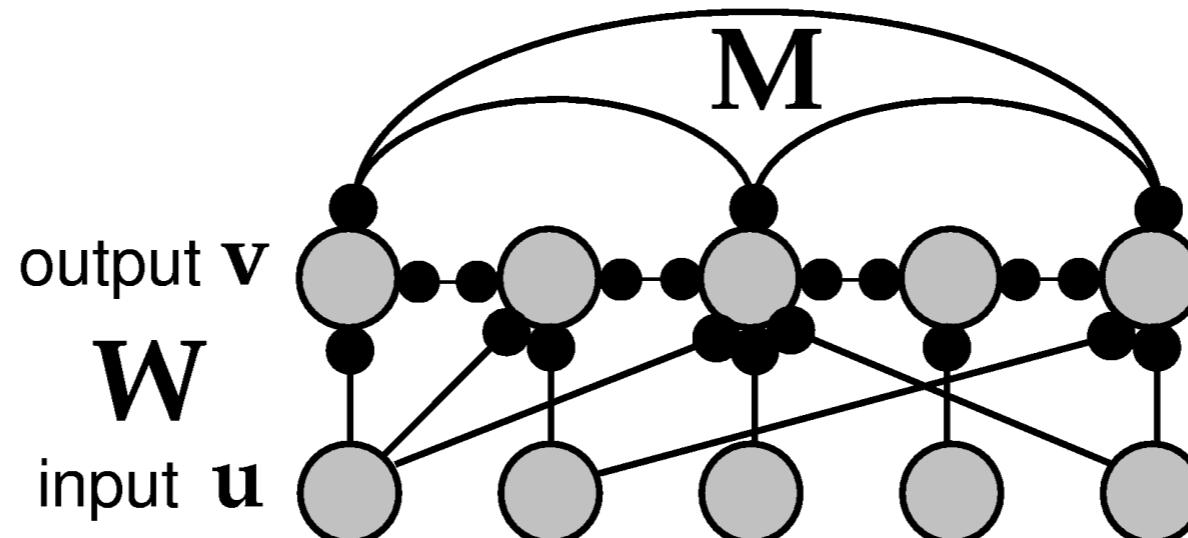
But it is more adequate to use *recurrent neural networks* to model time explicitly.



Dayan and Abbott book (2001)

# Recurrent neural networks

## mathematical models



**Discrete-time:**

[often used in machine learning]

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

$r$ , is the activity decay

**Continuous-time:**

[often used in computational neuroscience]

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

$\tau_r$ , 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:

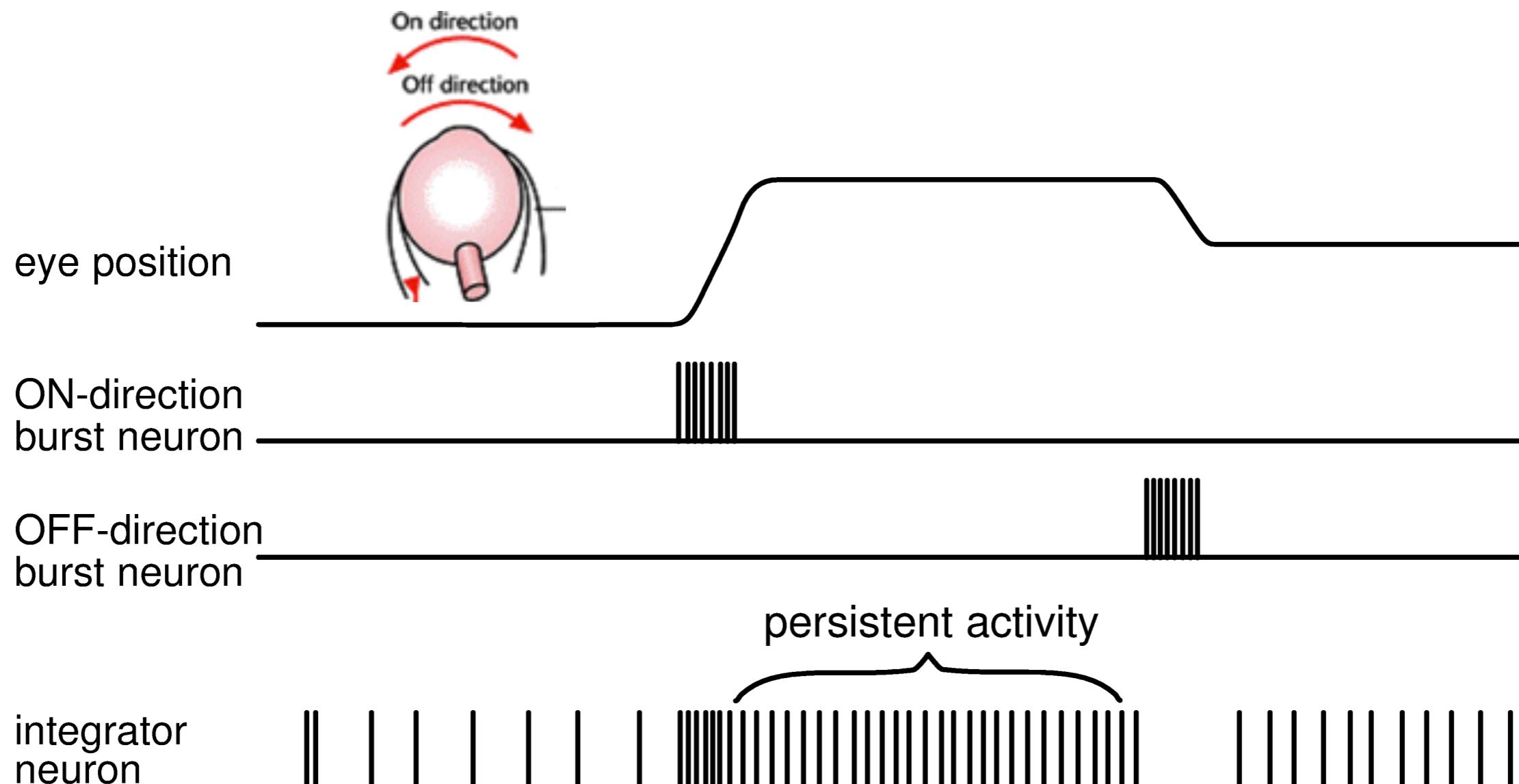


Seung et al, (2000)

# How does the brain remember eye position?

## Neural integrator for the oculomotor system

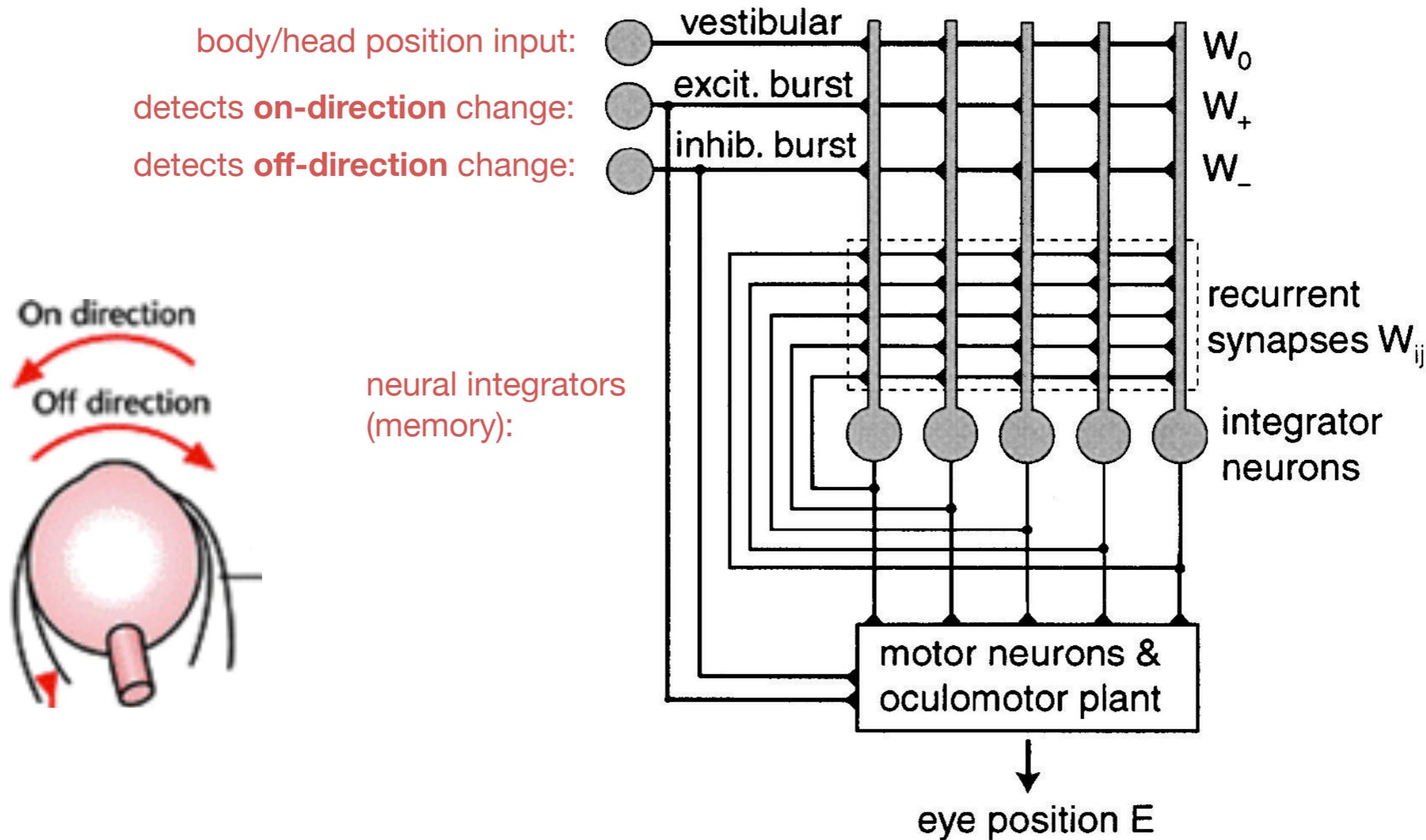
How does the brain remember where it is looking at?



Seung et al, Neuron (2000)

# How does the brain remember eye position?

## Neural integrator for the oculomotor system



# **Quiz time!**

**Please go to BB  
and solve quiz 6.**

**It should take you just a couple of minutes.**

# Neural integration in RNNs

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

*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  $\text{diag}(M) = 0$ , so no self recurrence

# Neural integration in RNNs

$r$  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 ‘forever’

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.

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

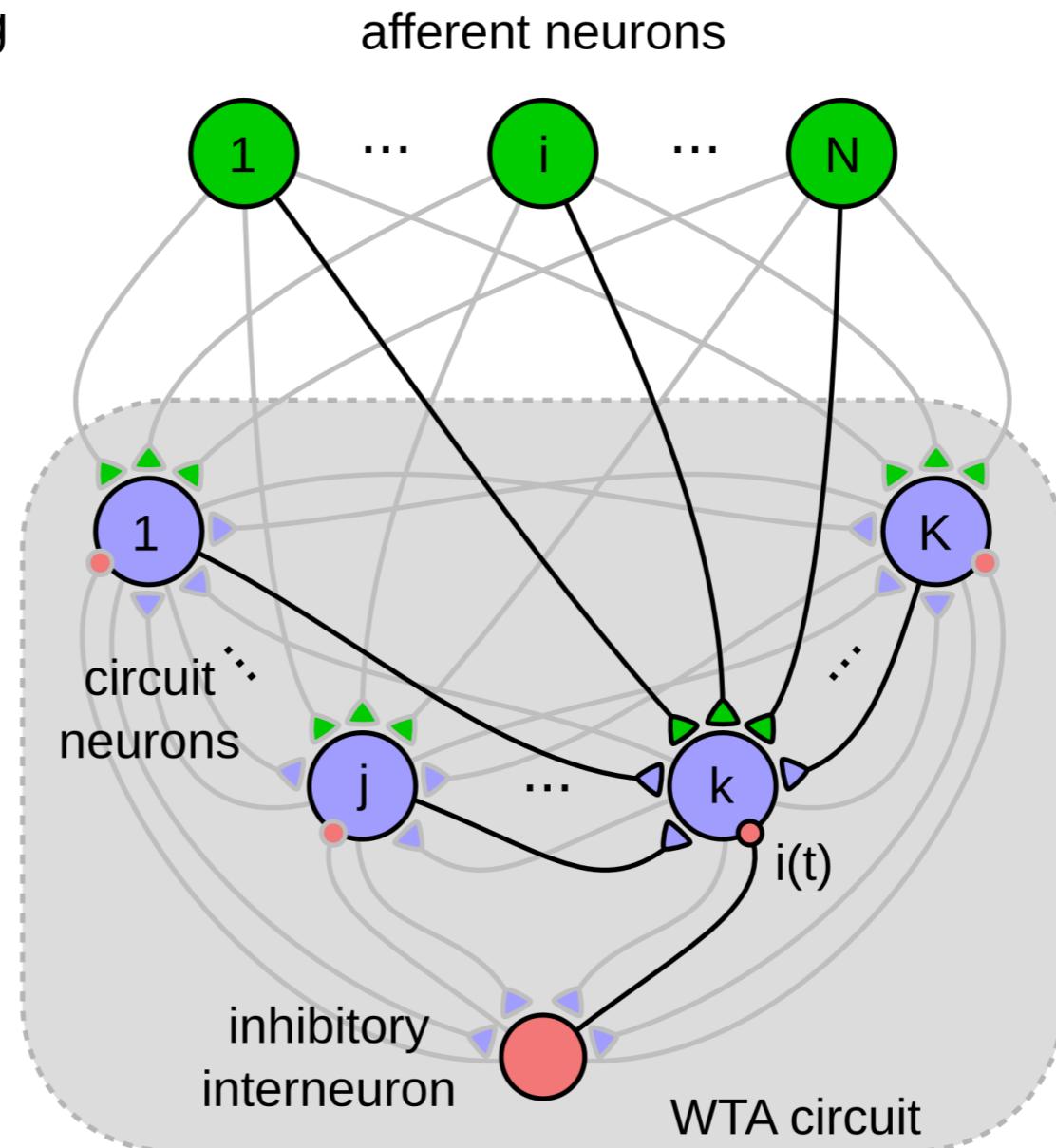
For simplicity, we assume here that  $\text{diag}(M) = 0$ , so no self recurrence

# Selecting only the strongest response

## Winner-take-all (WTA) network

**WTA operation** is similar to applying the *max* operator on the response of the  $k$  neurons (in purple).

Using lateral inhibitory neurons is a way of implementing such a max operation in a network:

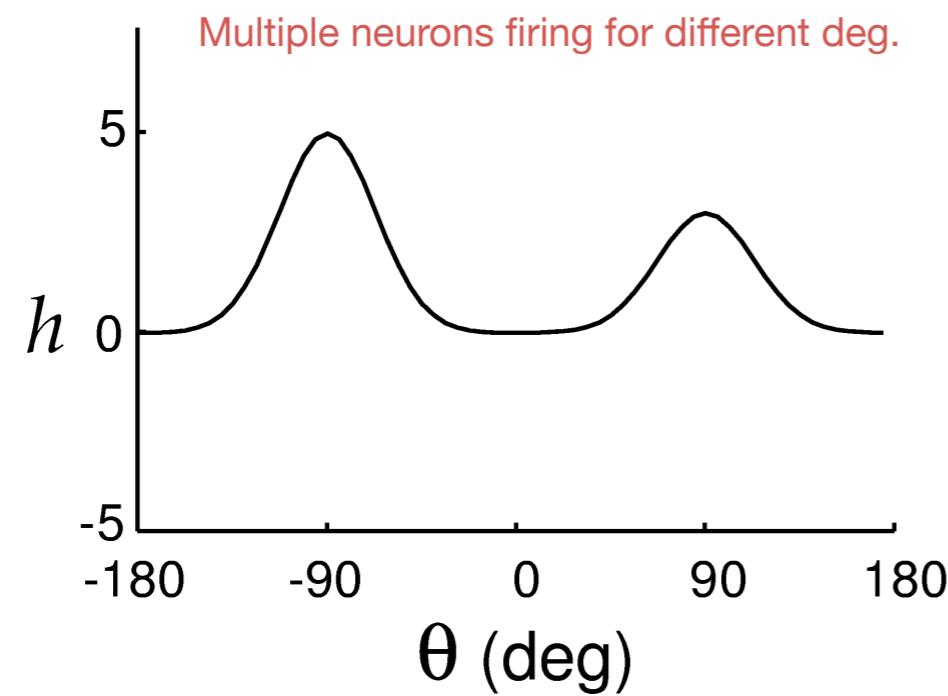


Kappel et al. PLoS CB (2014)

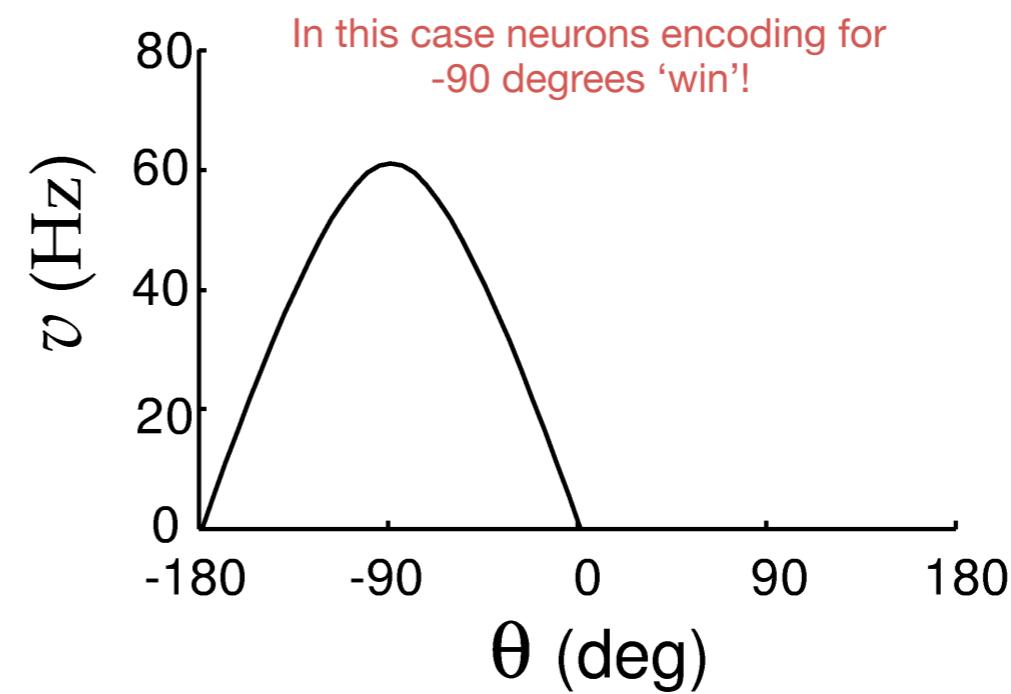
# Selecting only the strongest response

## Winner-take-all network

**Before winner-take-all**



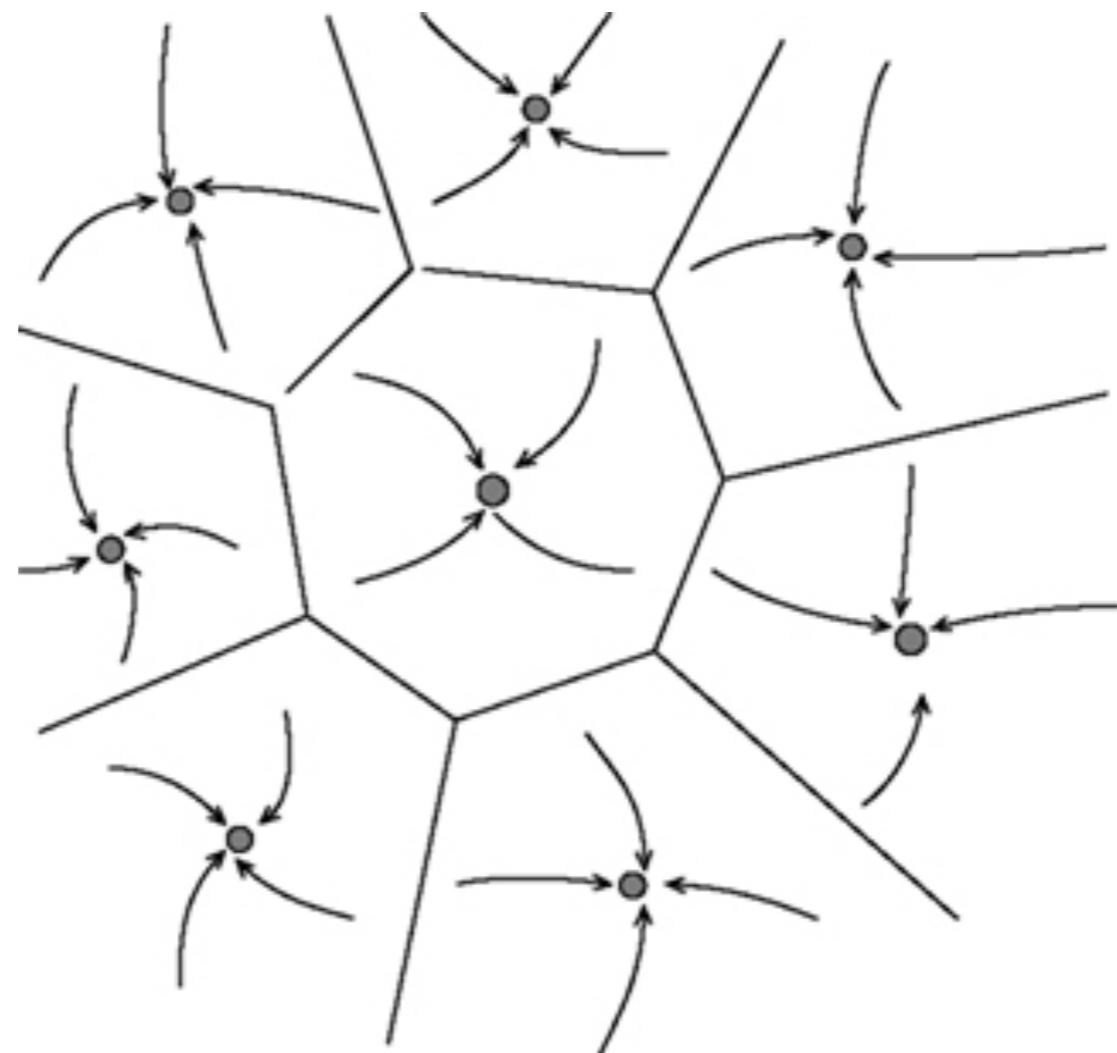
**After winner-take-all**



Dayan and Abbott book (2001)

# Attractor neural networks

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

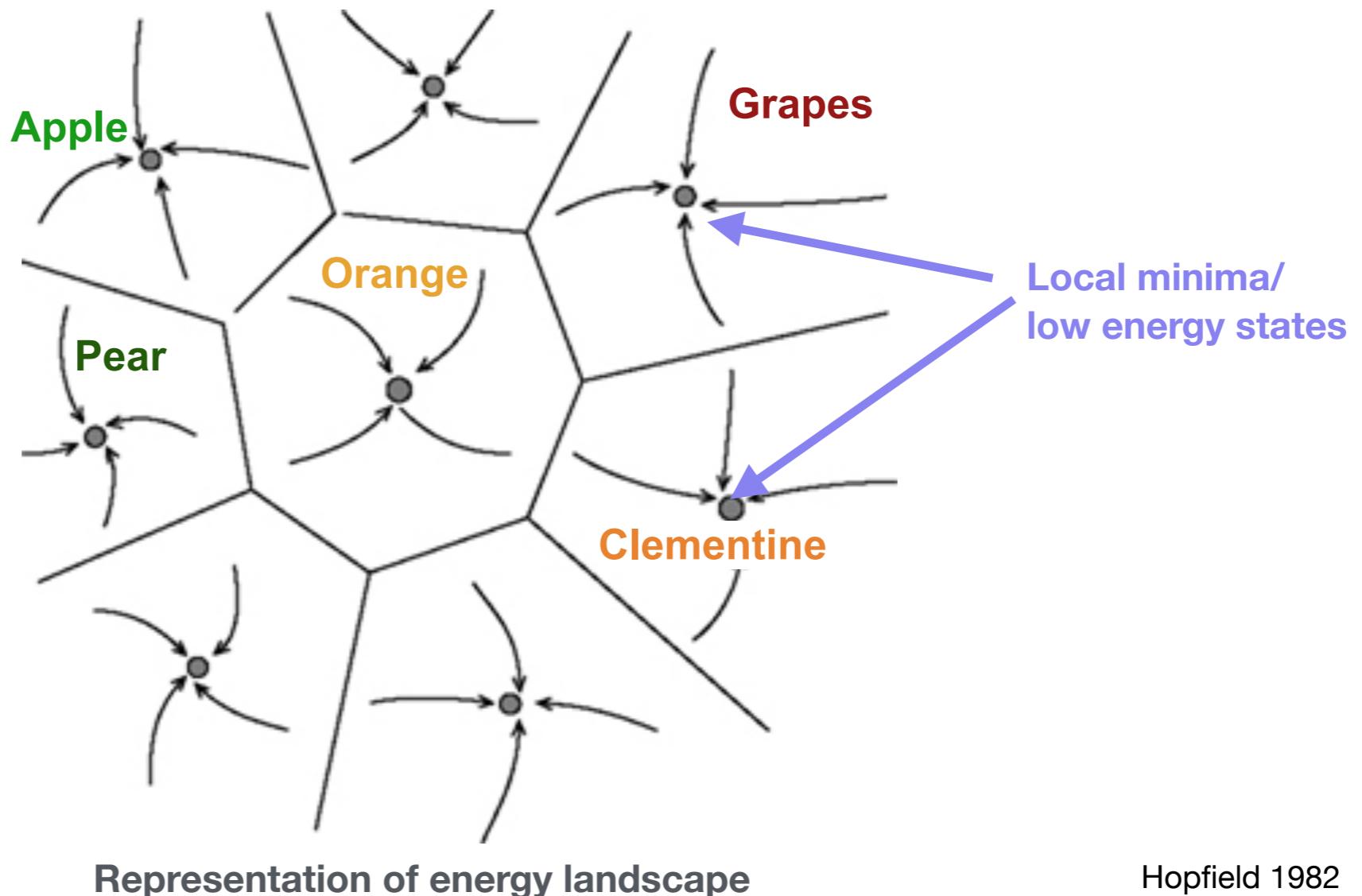


Dayan and Abbott book (2001)

# 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.



# Attractor neural networks

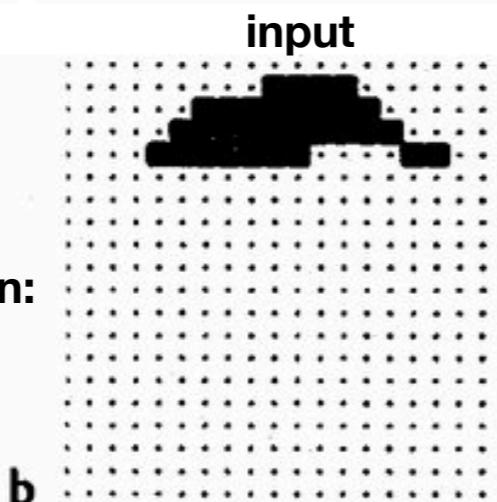
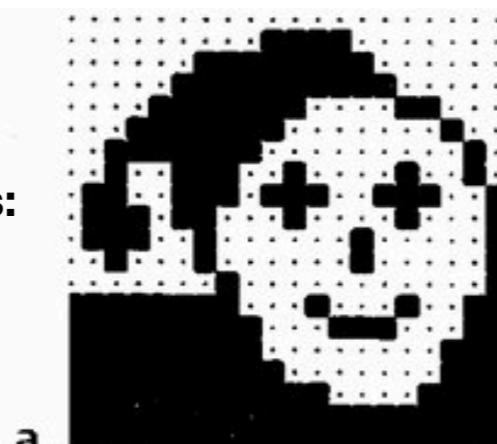
## Hopfield networks

**Two key properties:**

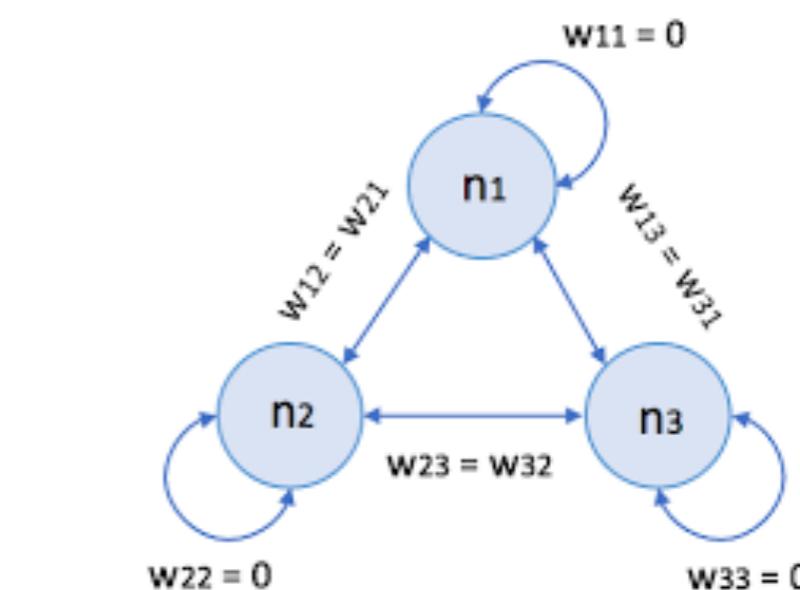
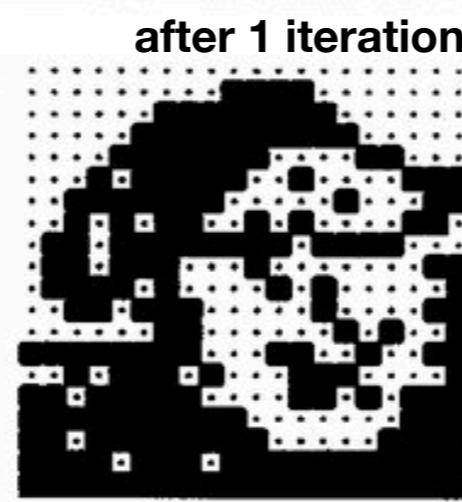
$m_{ii} = 0$  no unit has a connection with itself

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

**Trained patterns:**

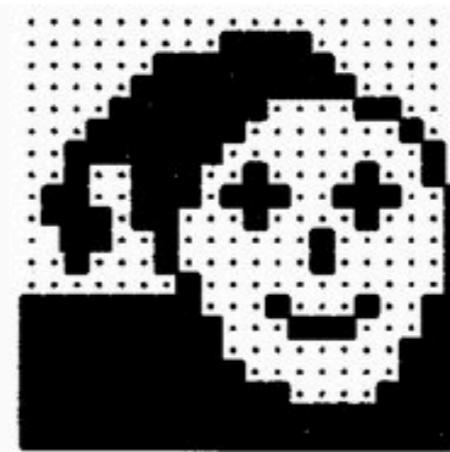


**Example of pattern completion:**



Hopfield connectivity,  $\mathbf{W}$  here is the same as  $\mathbf{M}$  in previous slides

**after 2 iterations**



Hopfield 1982

# Attractor neural networks

## Boltzmann machines

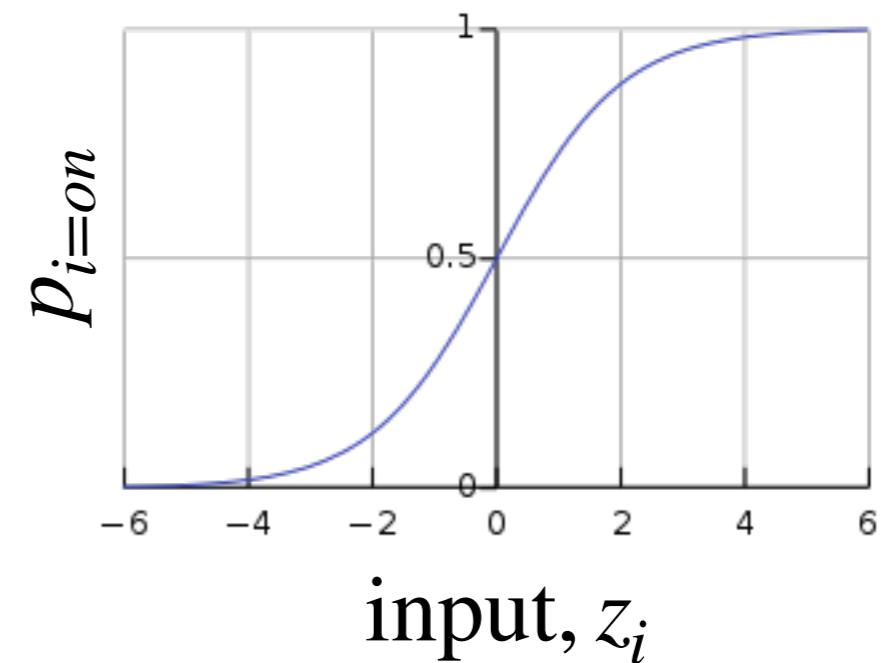
**Boltzmann machines** are the stochastic version of Hopfield networks.

Total input:

$$z_i = b_i + \sum_j \text{external input} w_{ij} + \sum_x \text{recurrent input} m_{ix}$$

Probability of turning on a given neuron  $i$ :

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



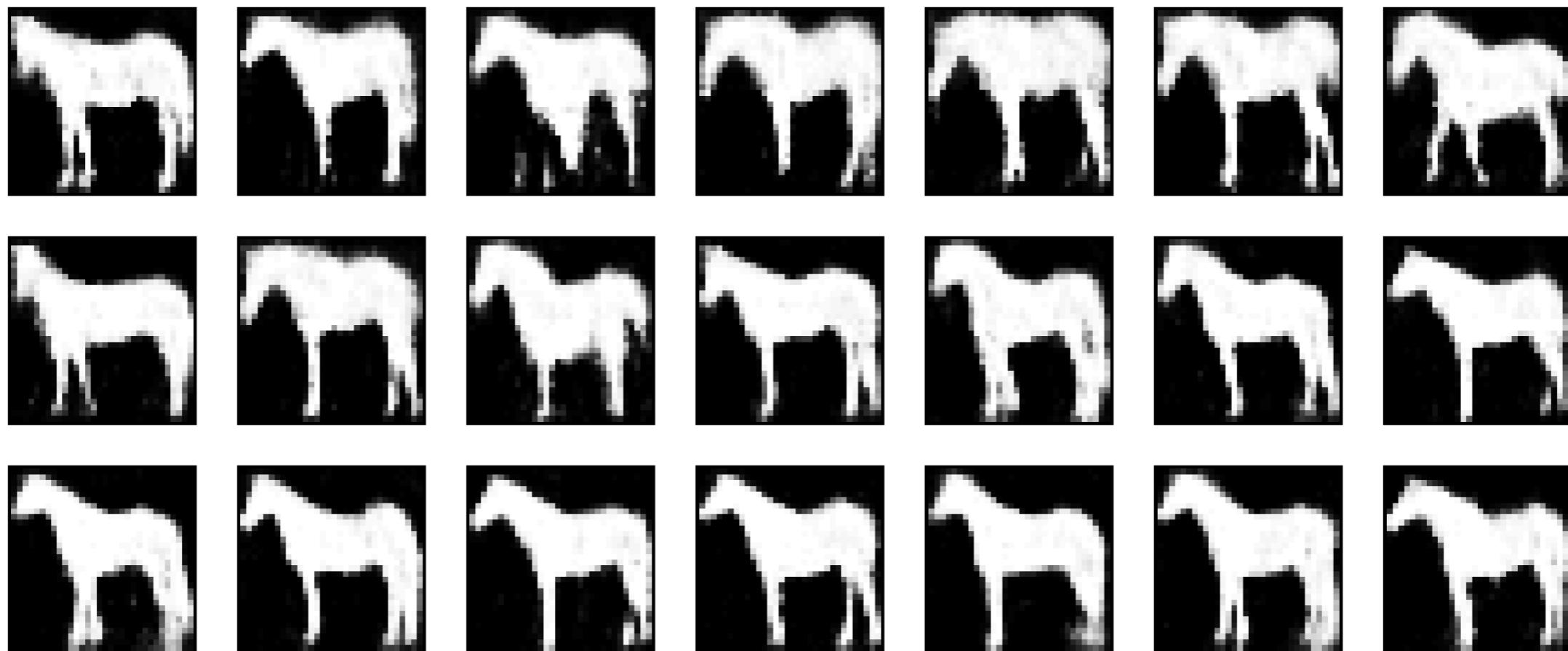
For temperature,  $T=0$  we obtain Hopfield networks.

[www.scholarpedia.org/article/Boltzmann\\_machine](http://www.scholarpedia.org/article/Boltzmann_machine)

# Autoassociative neural networks

## Boltzmann machines

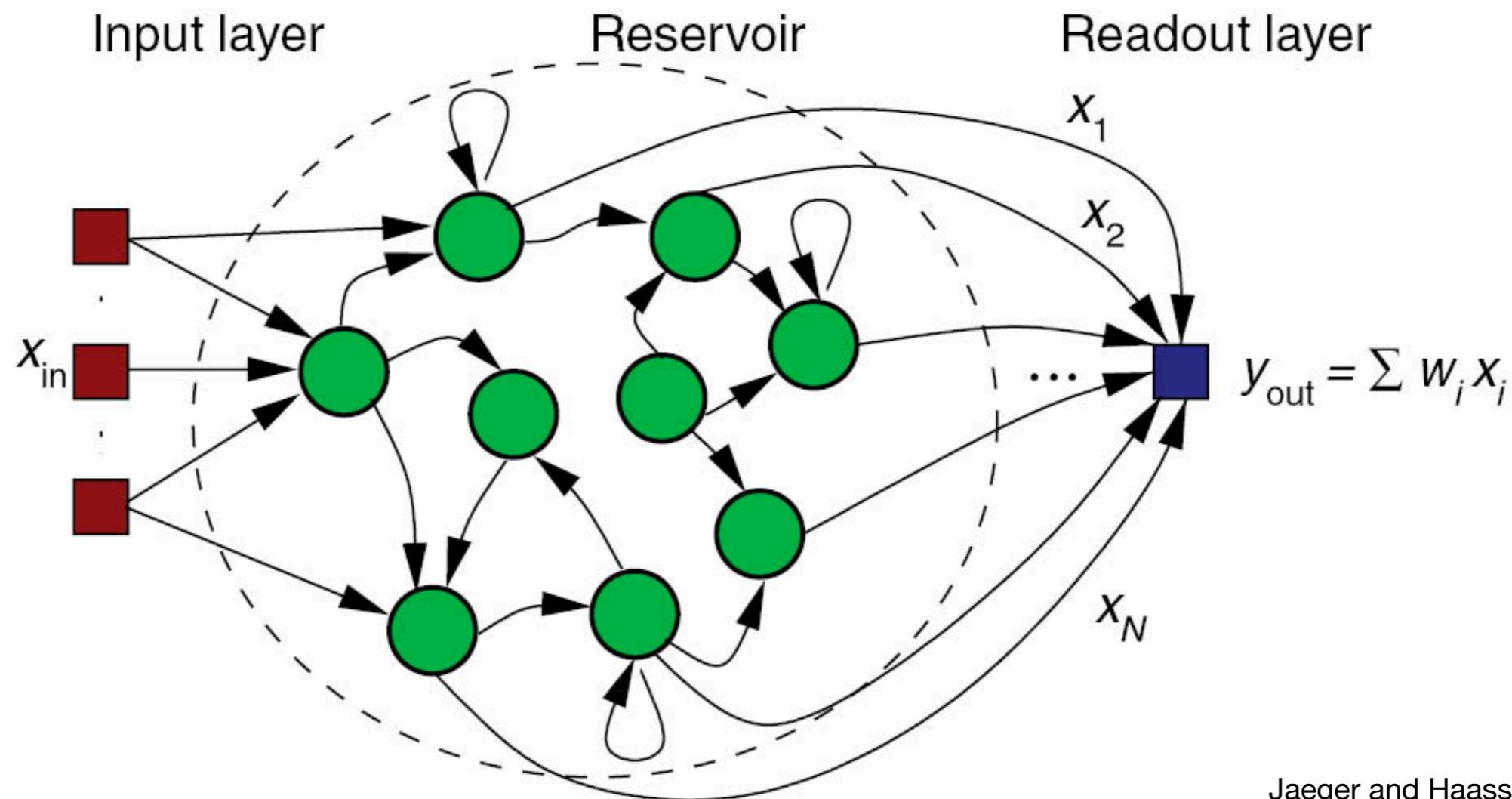
**Boltzmann machines** are good generative models. Samples from a Boltzmann machine after it learned the shape of horses:



Eslami et al. (2014)

# Reservoir computing

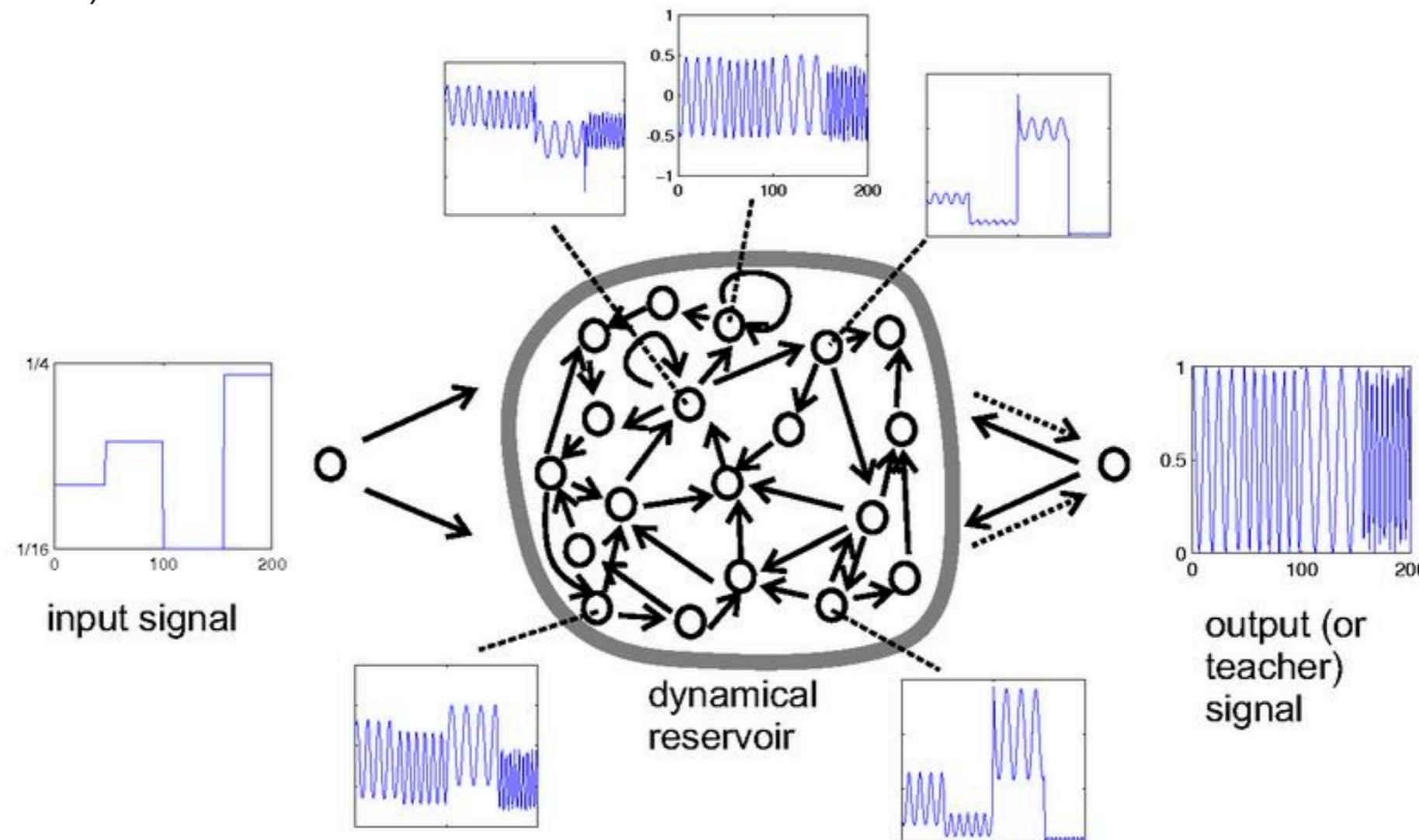
**Key idea:** Use a large pool of (sparsely connected) neurons which generate a large repertoire of dynamics (the reservoir with **fading memory**), and learn a simple read-out to combine the interesting dynamics into an useful output. It expands the input into higher dimensions to be easier to separate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



Jaeger and Haass 2004

# Reservoir computing: echo state networks

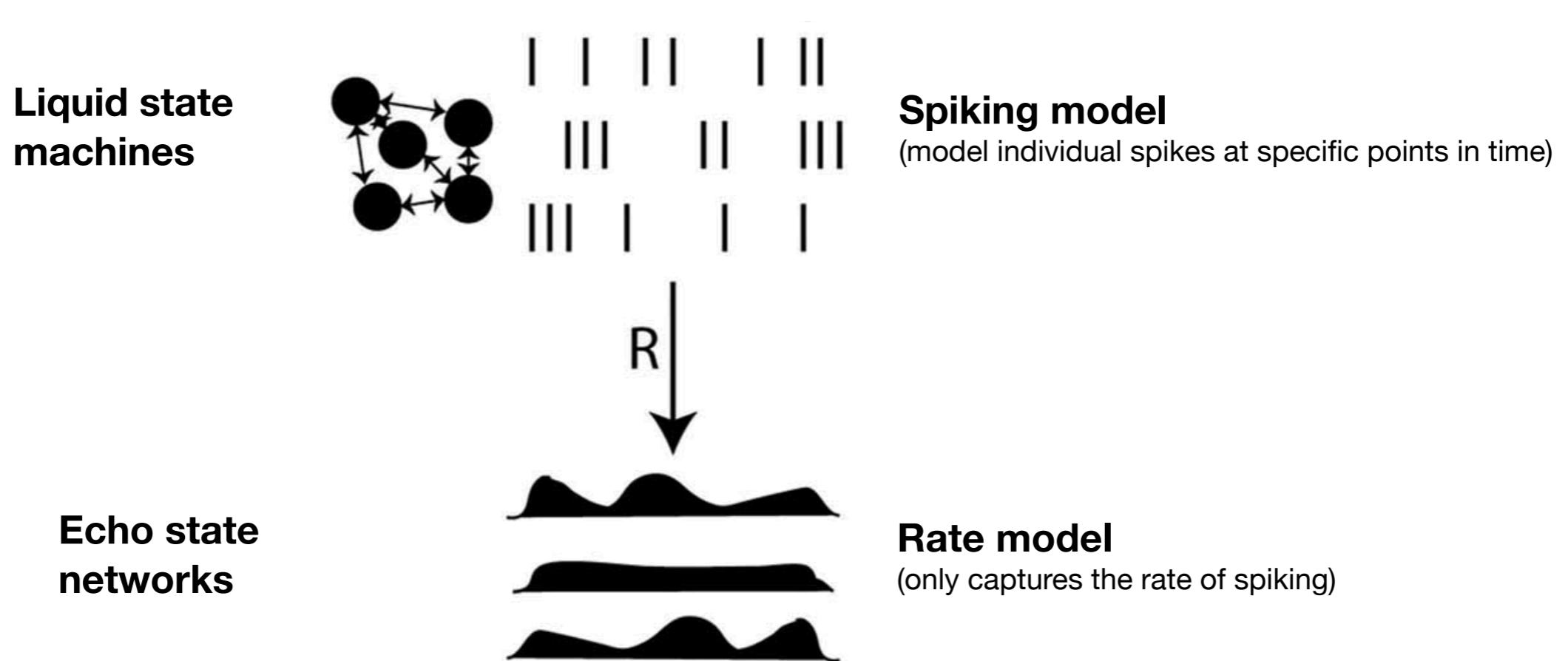
**Echo state networks:** is the most common rate-based (i.e. neurons are modelled at the level of firing rates) reservoir network.



Jaeger and Haass 2004

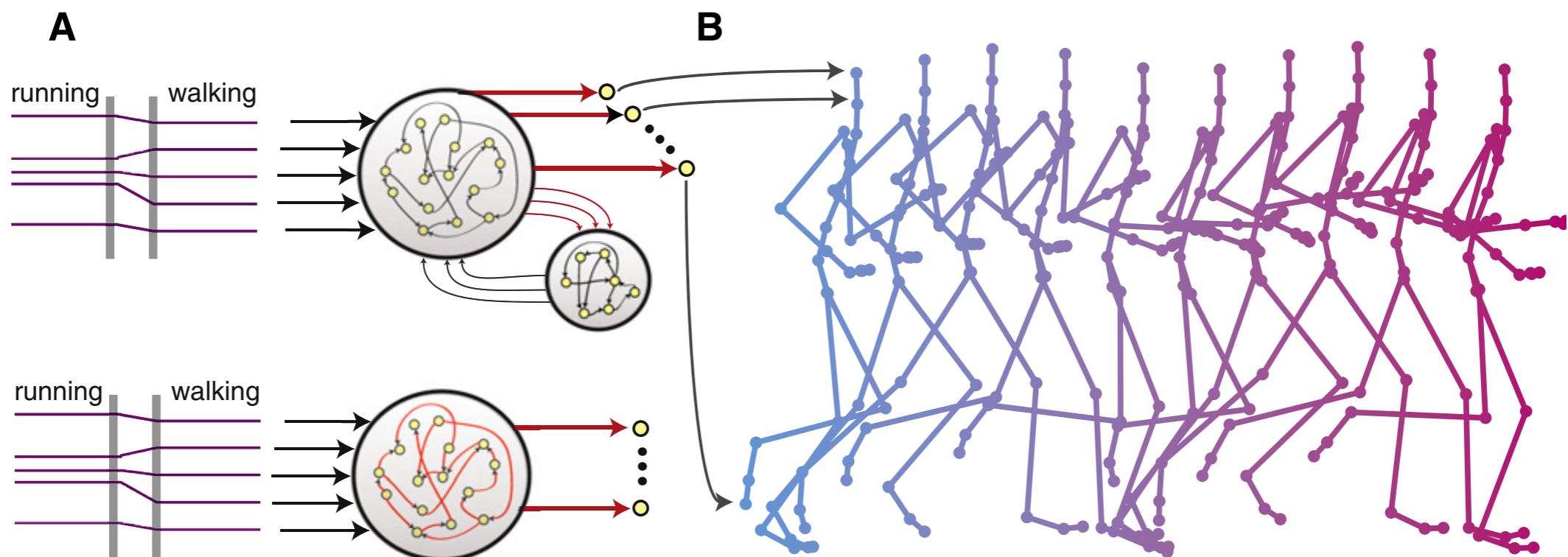
# Reservoir computing: liquid state machines

**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.



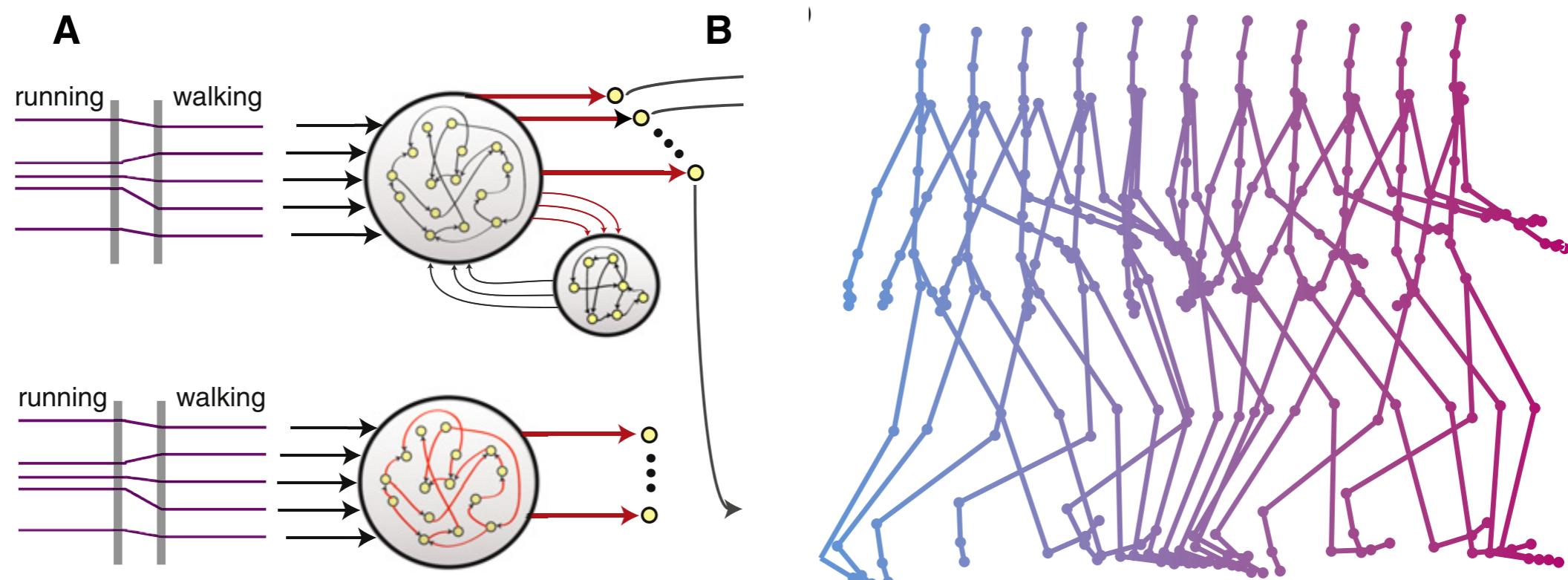
Maass et al. Neural Computation (2002)

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



Sussillo and Abbott Neuron (2009)

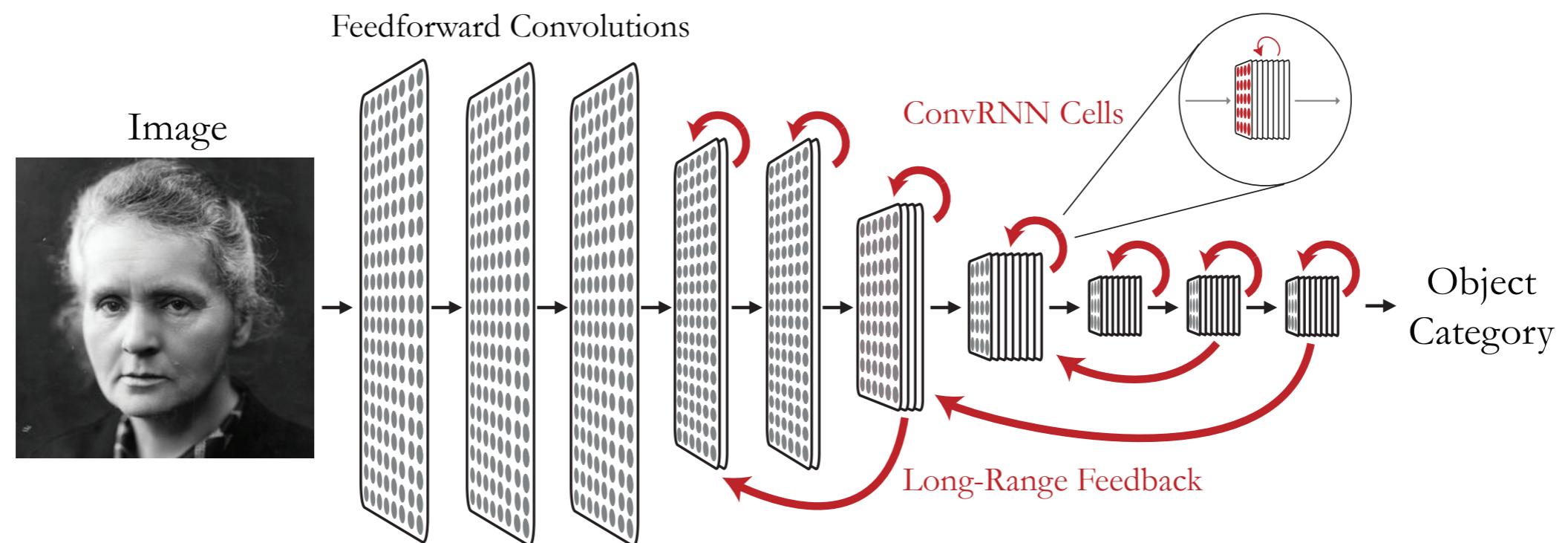
# Reservoir computing: a network that can learn to run/walk



Sussillo and Abbott Neuron (2009)

# Making CNNs recurrent

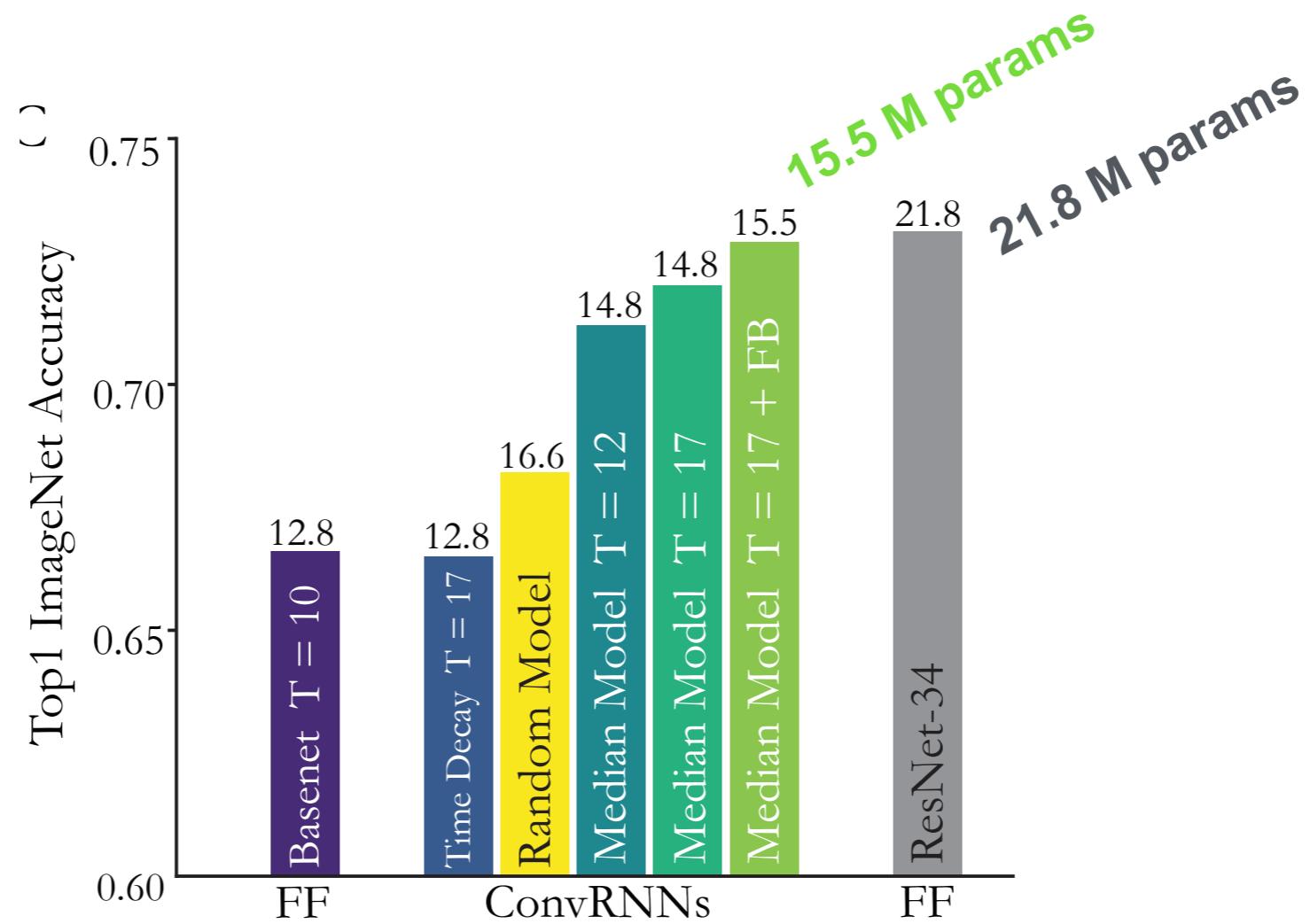
Most brain areas exhibit a high degree of recurrence (both within and between brain areas), recent developments show that if convolutional neural networks (CNNs) are extended with specific forms of recurrence they can outperform purely feedforward CNNs.



Nayebi et al. NeurIPS (2018)

# Making CNNs recurrent

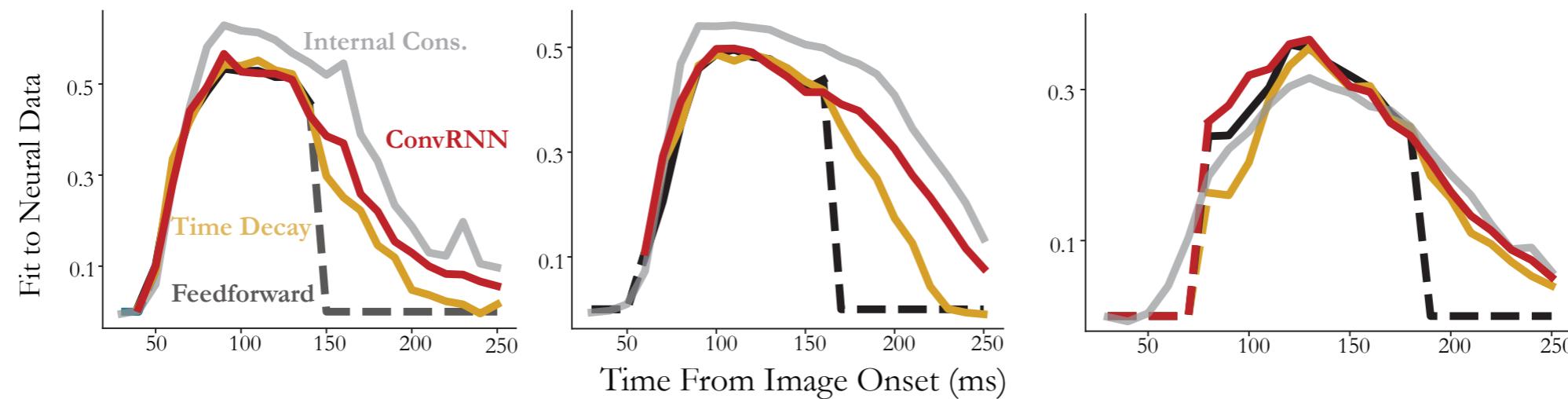
Nayebi et al. searched over multiple forms of recurrent CNNs (ConvRNNs) and found that they can still obtain good performance, while using *less depth* as current deep networks, and *less parameters*.



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.



The gradual rise and decay in ConvRNNs is similar to how neurons respond in time when presented with a stimuli.

Nayebi et al. NeurIPS (2018)

# Summary

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

# Upcoming lectures

- L1<sup>[4]</sup>: Neural circuits and learning: introduction
- L2<sup>[4]</sup>: Supervised learning & backprop
- L3<sup>[5]</sup>: Visual system: deep learning?
- L4<sup>[5]</sup>: Reinforcement learning
- L5<sup>[6]</sup>: Unsupervised learning
- L6<sup>[6]</sup>: Temporal processing
- L7<sup>[7]</sup>: **RNNs and Brain vs maquina**
- L8<sup>[7]</sup>: Guest lecture