NSC3270 / NSC5270
Computational Neuroscience

Tu/Th 9:35-10:50am
Featheringill Hall 129

Professor Thomas Palmeri
Professor Sean Polyn

For Today

Required Video

But what *is* a Neural Network?

In-Class Python Code:

`ActivFuncs.py` or `ActivFuncs.ipynb`

For Thursday

Required Readings

Chapter 3 (selected pages) of Churchland, P.S., & Sejnowski, T.J. (2017). *The Computational Brain* (25th Anniversary Edition). MIT Press.

links on Brightspace

At What Conceptual Levels of Analysis to Model?

Computational Level

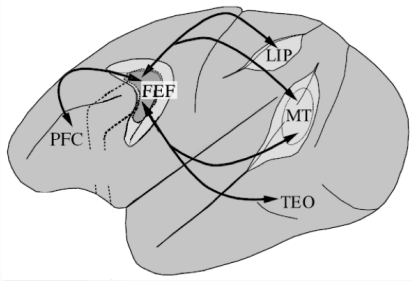
NORMATIVE

what are the goals of the organism
what is optimal for survival

Algorithmic/ Representational Level

SOFTWARE

what kinds of representations and computations are performed



Implementation/ Biophysical Level

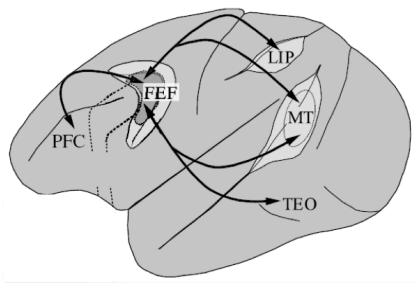
HARDWARE

how mechanisms are physically realized within a biological substrate (neurons and their connections)

At What Conceptual Levels of Analysis to Model?

Computational Level

Algorithmic/ Representational Level



Implementation/ Biophysical Level

BOTTOM UP APPROACH

how mechanisms are physically realized within a biological substrate (neurons and their connections)

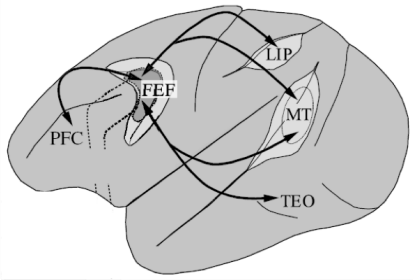
At What Conceptual Levels of Analysis to Model?

Computational Level

TOP DOWN APPROACH

what are the goals of the organism
what is optimal for survival

Algorithmic/ Representational Level

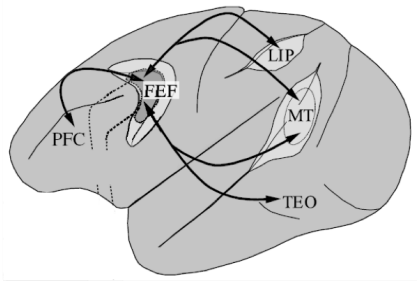


Implementation/ Biophysical Level

At What Conceptual Levels of Analysis to Model?

Computational Level

Algorithmic/ Representational Level



Implementation/ Biophysical Level

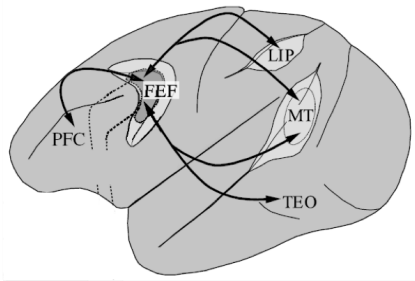
INSIDE OUT APPROACH

what kinds of representations and computations are performed

At What Conceptual Levels of Analysis to Model?

Computational Level

Algorithmic/ Representational Level



Implementation/ Biophysical Level

Economics

Psychology

Cognitive Neuroscience

Systems Neuroscience

Molecular Neuroscience

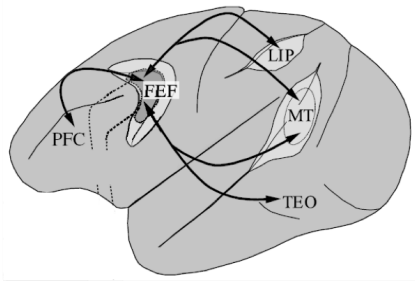
At What Conceptual Levels of Analysis to Model?

Computational Level

What features is most informative in a large collection of images?
optimality, statistics, information theory

Algorithmic/ Representational Level

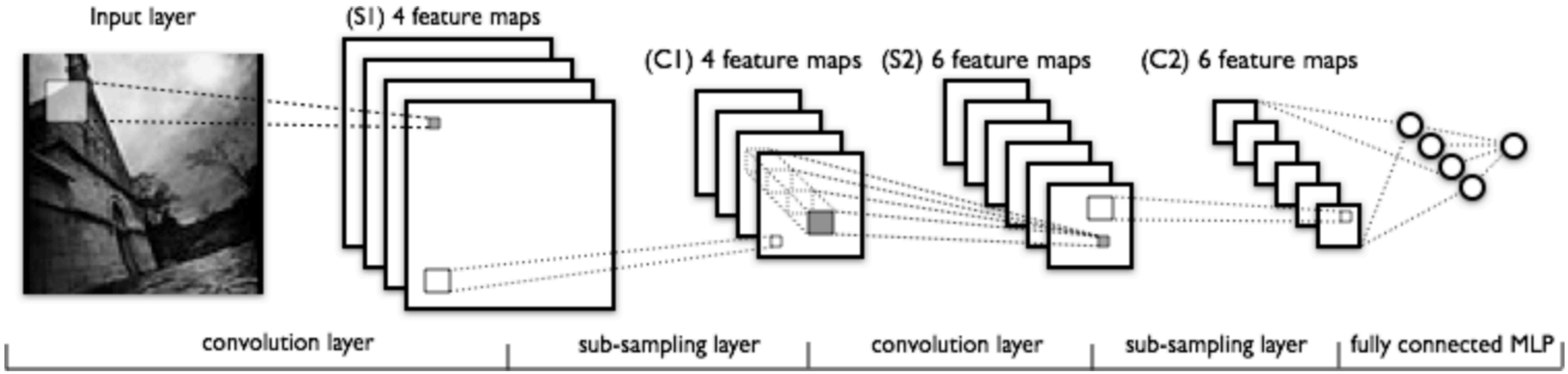
2nd derivative of a Gaussian smoothed image highlights edges in the image
image and signal processing



Implementation/ Biophysical Level

how do certain neurons in primary visual cortex (area V1) operate
spiking neural network models

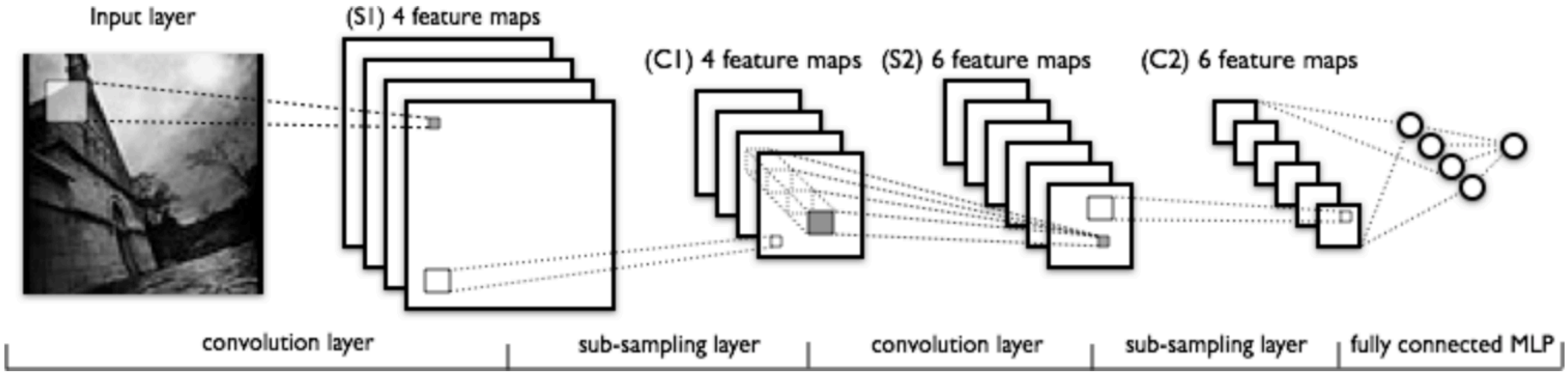
Example Neural Network Model



LeNet-5

http://yann.lecun.com/exdb/lenet/

Example Neural Network Model



LeNet-5

http://yann.lecun.com/exdb/lenet/

- our focus is on using neural network models to explain behavior and the brain
- these models are now used in devices - speech recognition, image recognition, self-driving cars
- and used to analyze big, complex data

in Toronto, [Hinton] works out of Google's office downtown, which is filled with advertising employees. He's the only researcher. Occasionally, a curious employee sidles up and asks, "What do you do?"

"Do you have an Android phone?" Hinton replies.

"Yes."

"The speech recognition is pretty good, isn't it?"

"Yes."

"Well, I design the neural networks that recognize what you say."

The questioner nearly always pauses in thought.

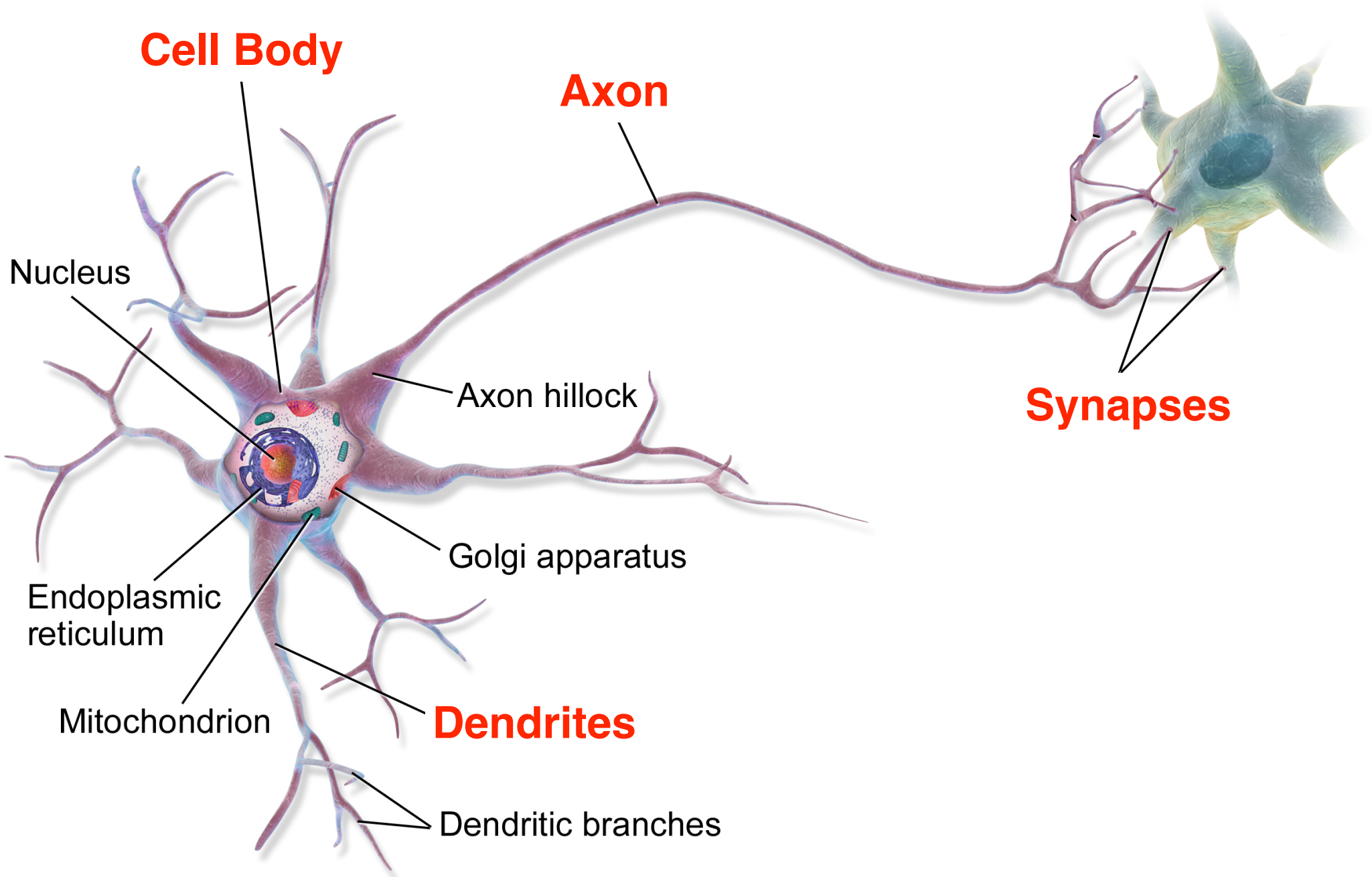
"Wait, what do you mean?"

from *The Believers: The hidden story behind the code that runs our lives*
by Paul Voosen, The Chronicle of Higher Education, Feb 23, 2015

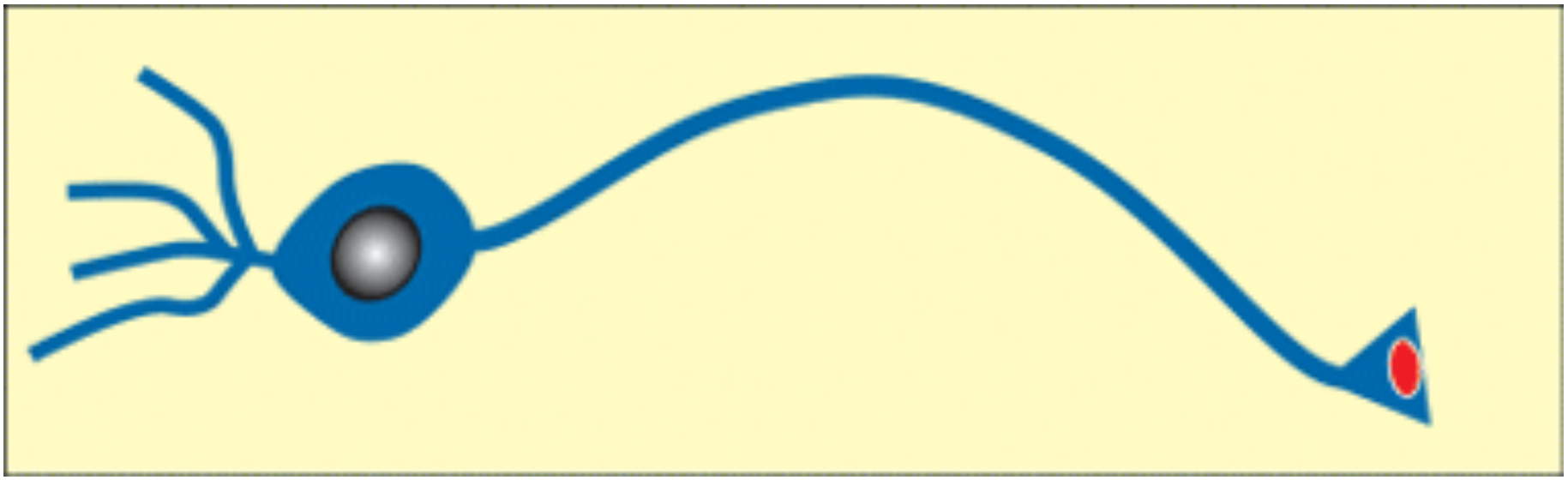
Simple Models of Neurons

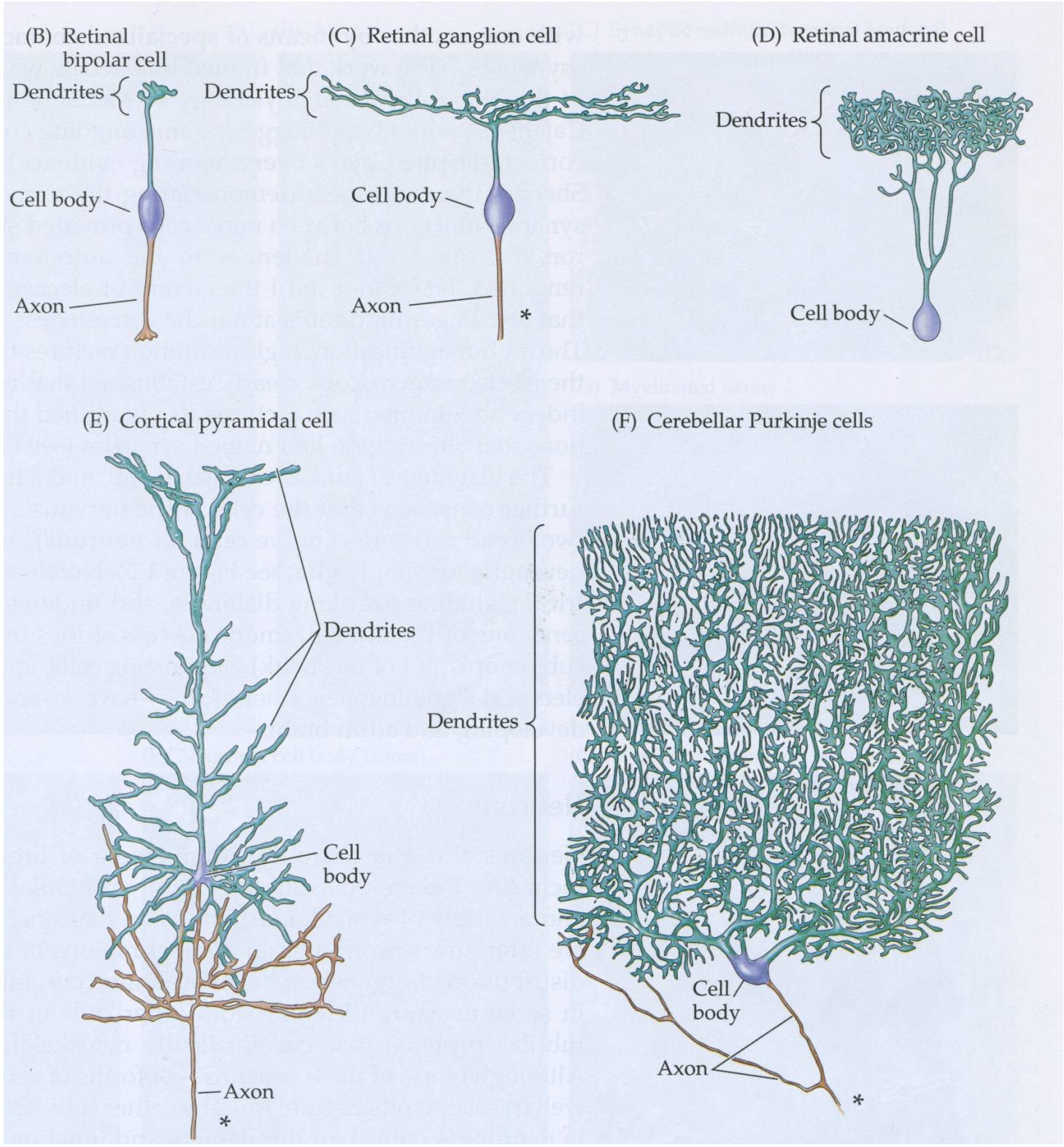
brief overview/review of neuron structure and function

Neuron

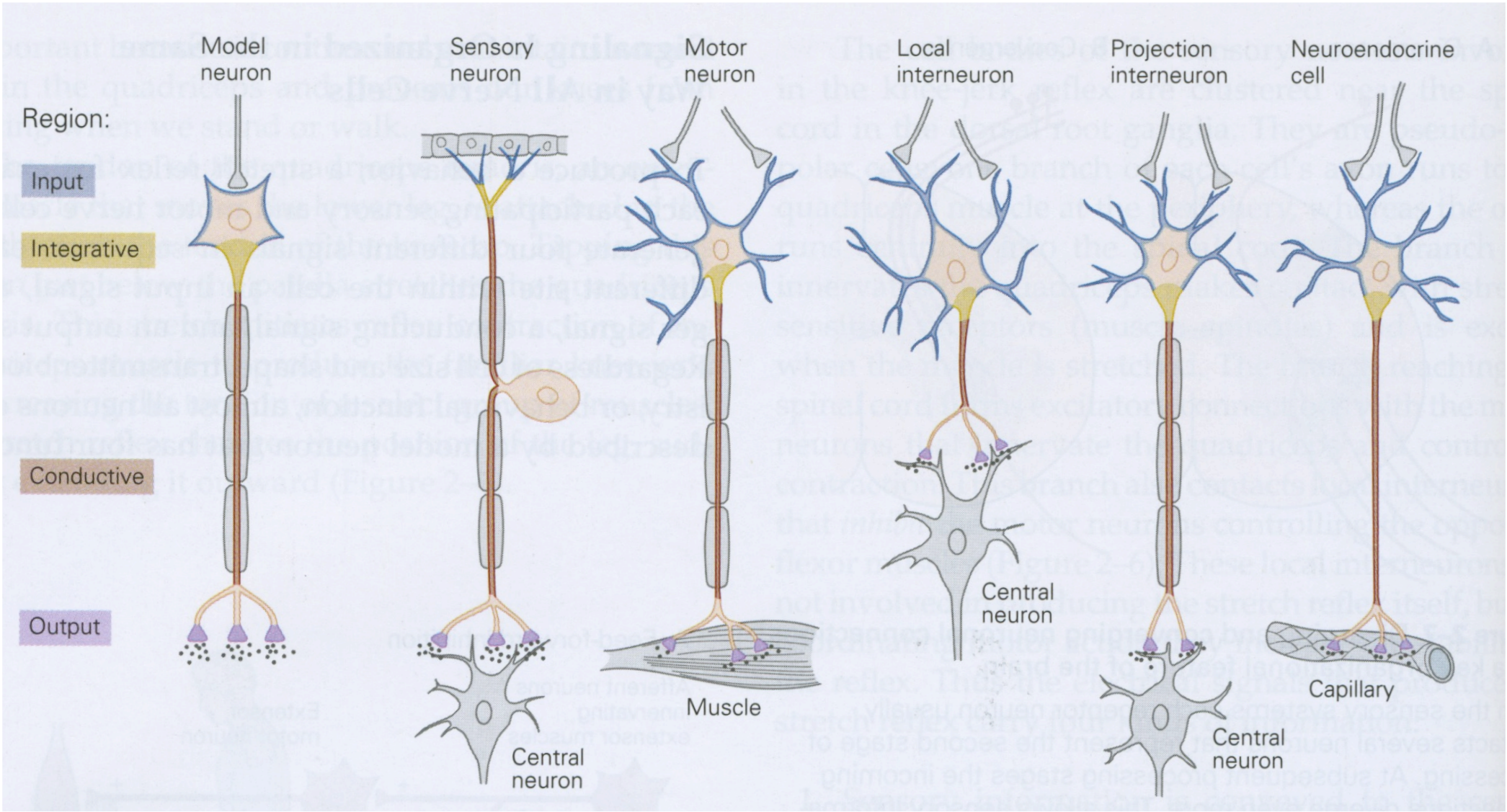

Cell Body
Axon
Nucleus
Axon hillock
Synapses
Golgi apparatus
Endoplasmic reticulum
Mitochondrion
Dendrites
Dendritic branches

Dendrites Cell Body Axon Synapses

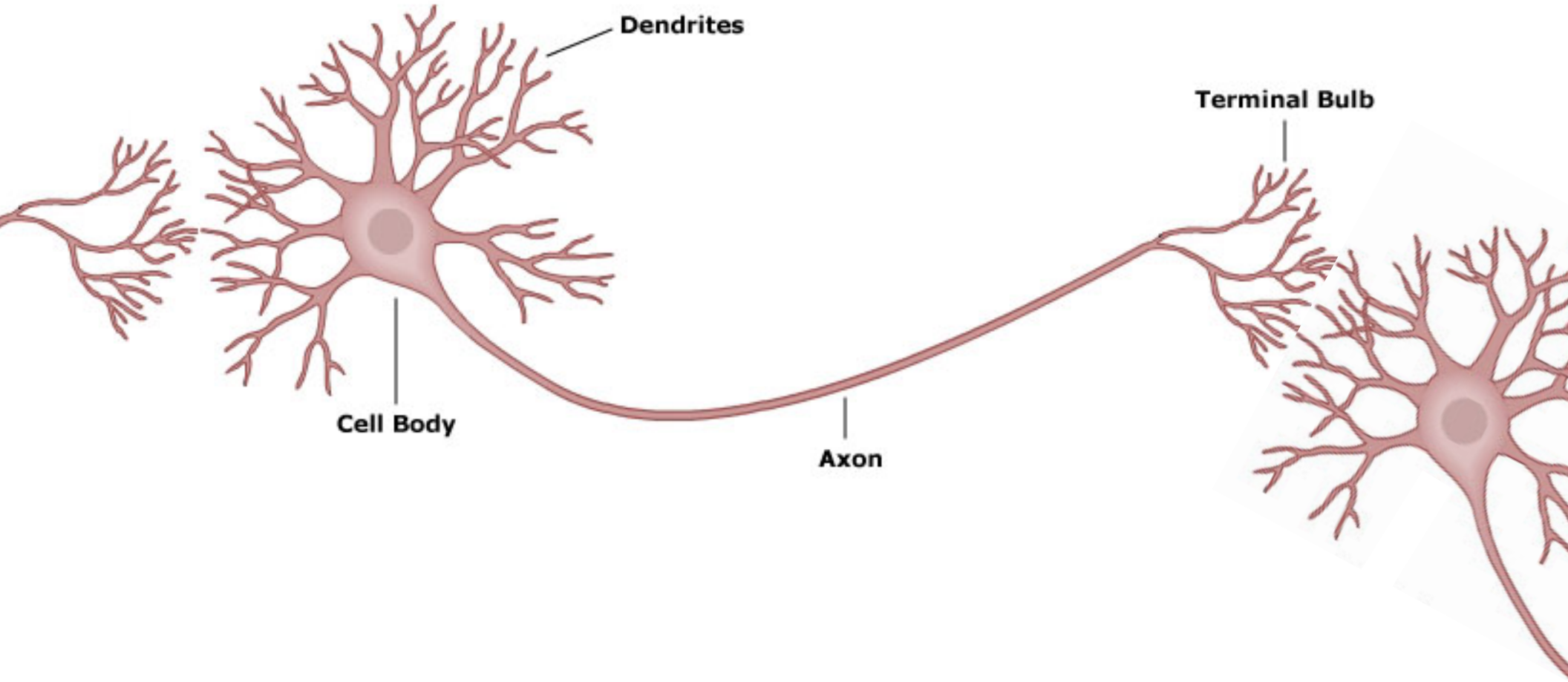


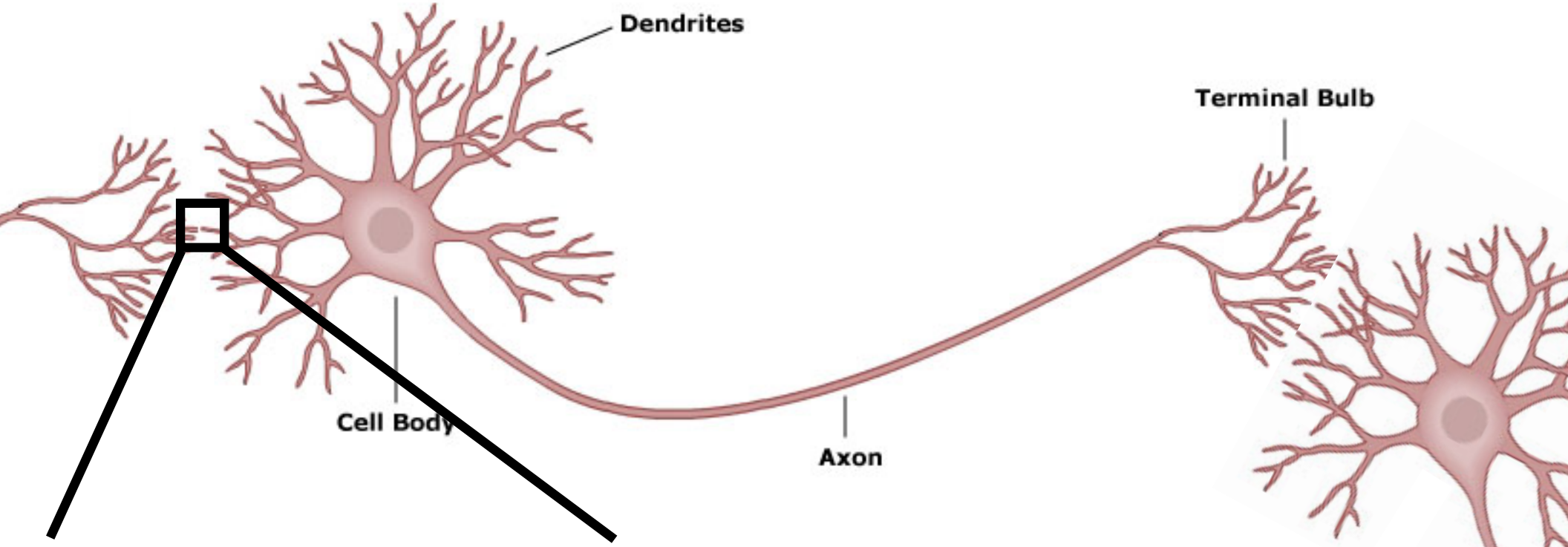

(B) Retinal bipolar cell
Dendrites
Cell body
Axon
(C) Retinal ganglion cell
Dendrites
Cell body
Axon
*
(D) Retinal amacrine cell
Dendrites
Cell body
(E) Cortical pyramidal cell
Dendrites
Cell body
Axon
*
(F) Cerebellar Purkinje cells
Dendrites
Cell body
Axon
*

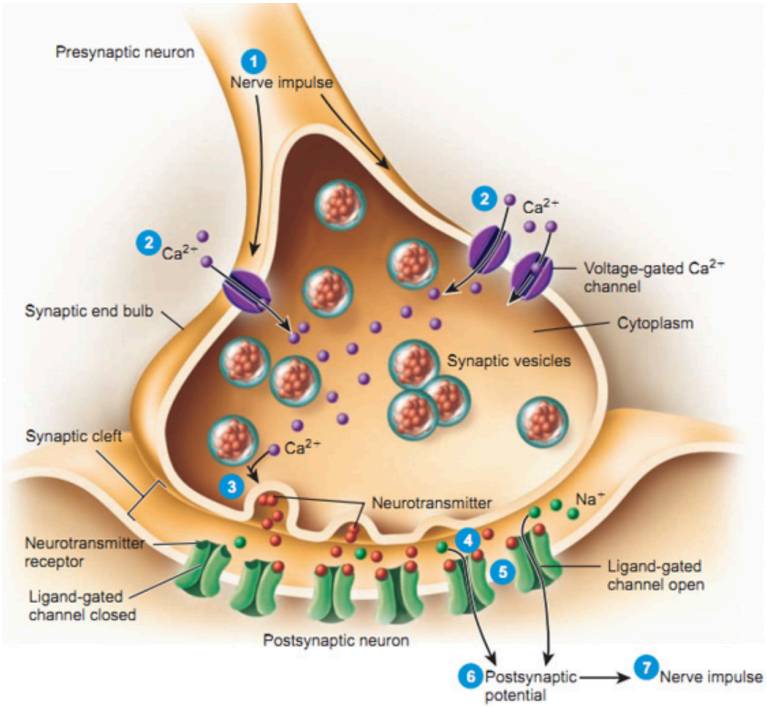
functional similarities despite morphological differences

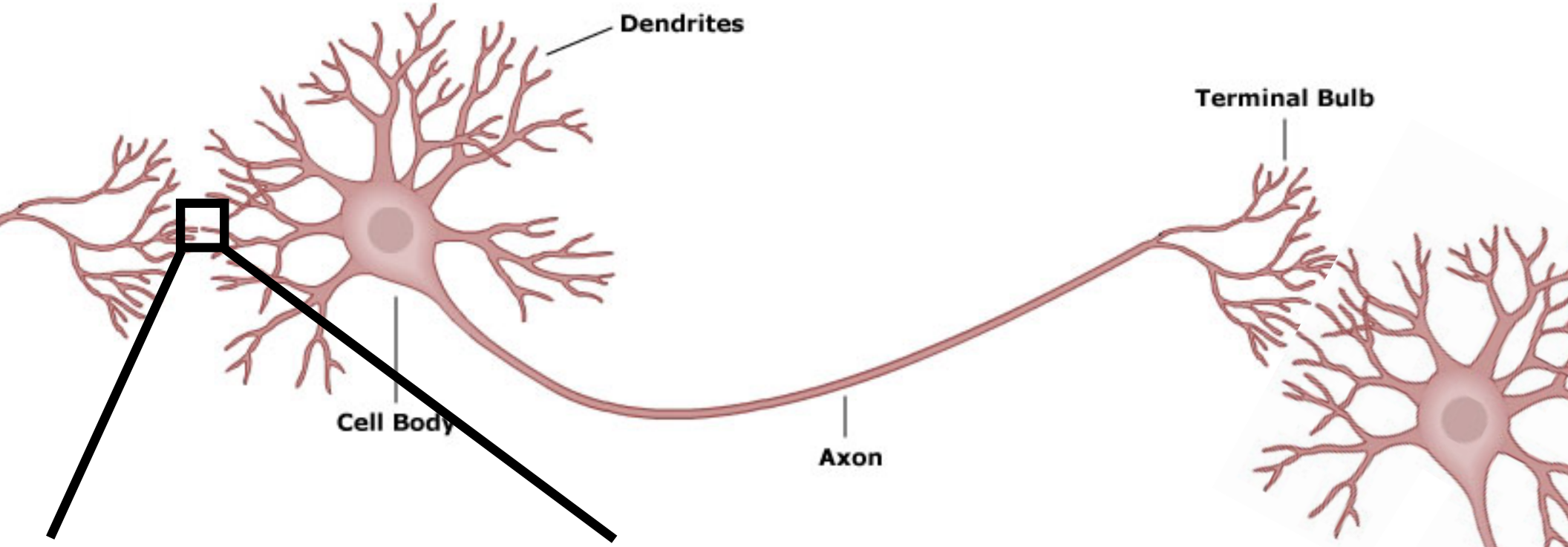


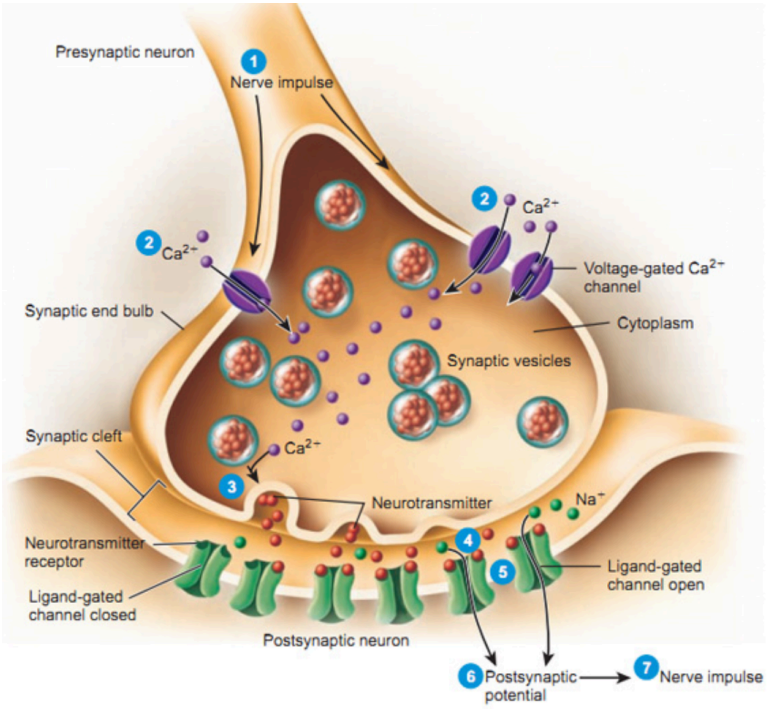
we will not discuss different types of neurons,
but instead model idealized neurons


Dendrites
Terminal Bulb
Cell Body
Axon


Dendrites
Terminal Bulb
Cell Body
Axon

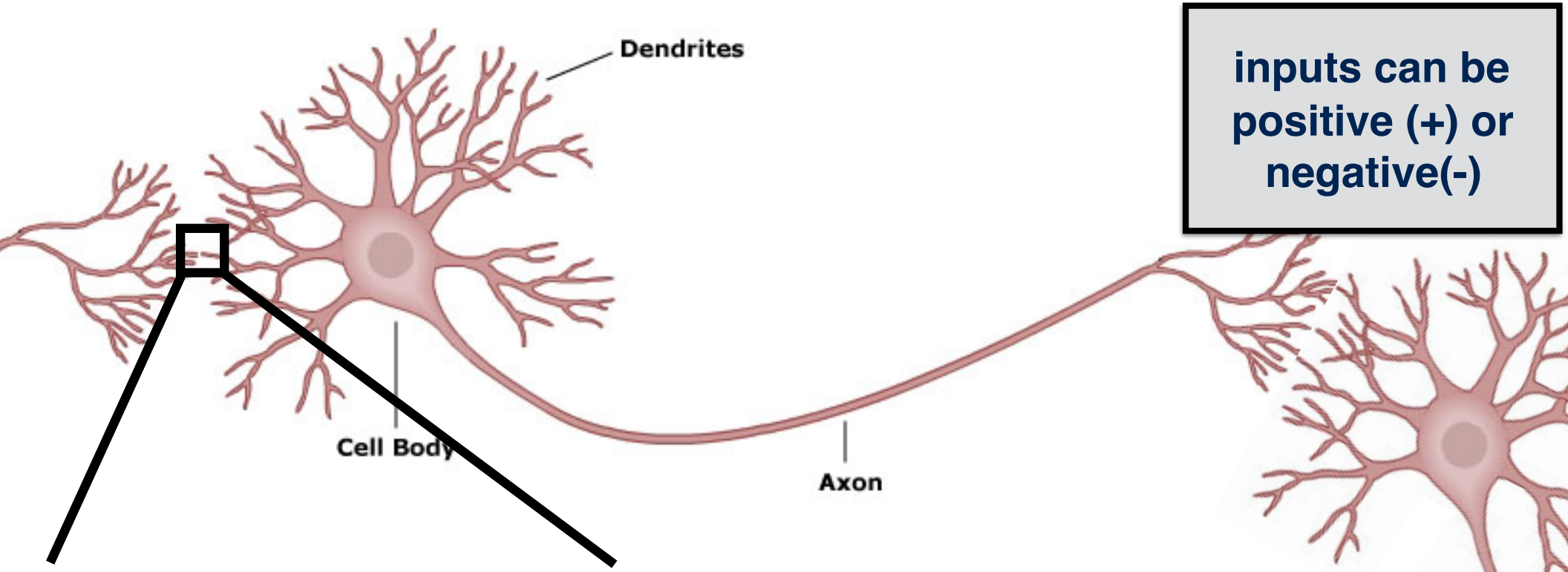

Presynaptic neuron
1
Nerve impulse
2
Ca2+
2
Ca2+
Voltage-gated Ca2+ channel
Synaptic end bulb
Cytoplasm
Synaptic vesicles
Synaptic cleft
Ca2+
3
Neurotransmitter
Na+
Neurotransmitter receptor
4
5
Ligand-gated channel open
Ligand-gated channel closed
Postsynaptic neuron
6
Postsynaptic potential
7
Nerve impulse



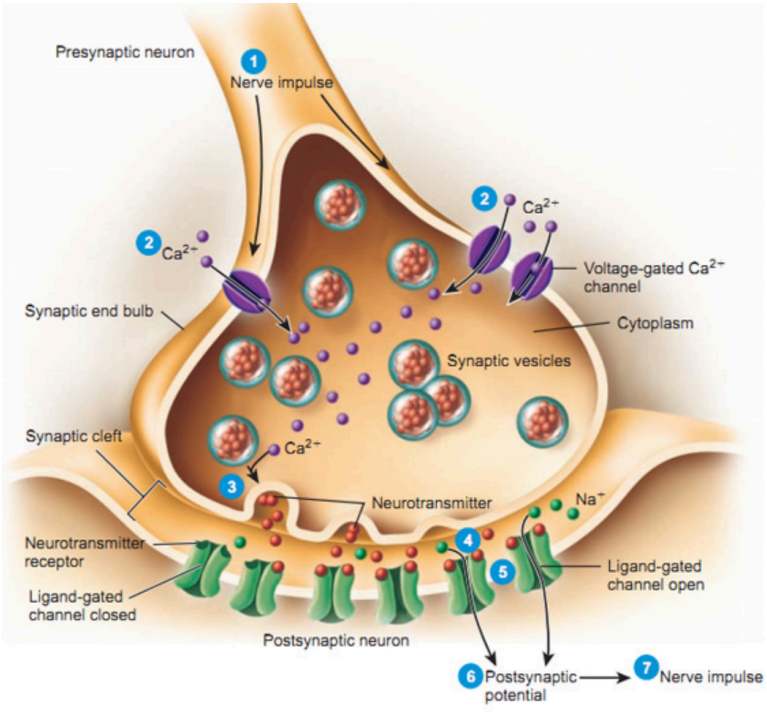


EPSPs (excitatory postsynaptic potentials) positive changes in membrane potential caused by excitatory pre-synaptic neurons (e.g., glutamate and NMDA receptors)

IPSPs (inhibitory postsynaptic potentials) negative changes in membrane potential caused by inhibitory pre-synaptic neurons (e.g., GABA and GABA receptors)

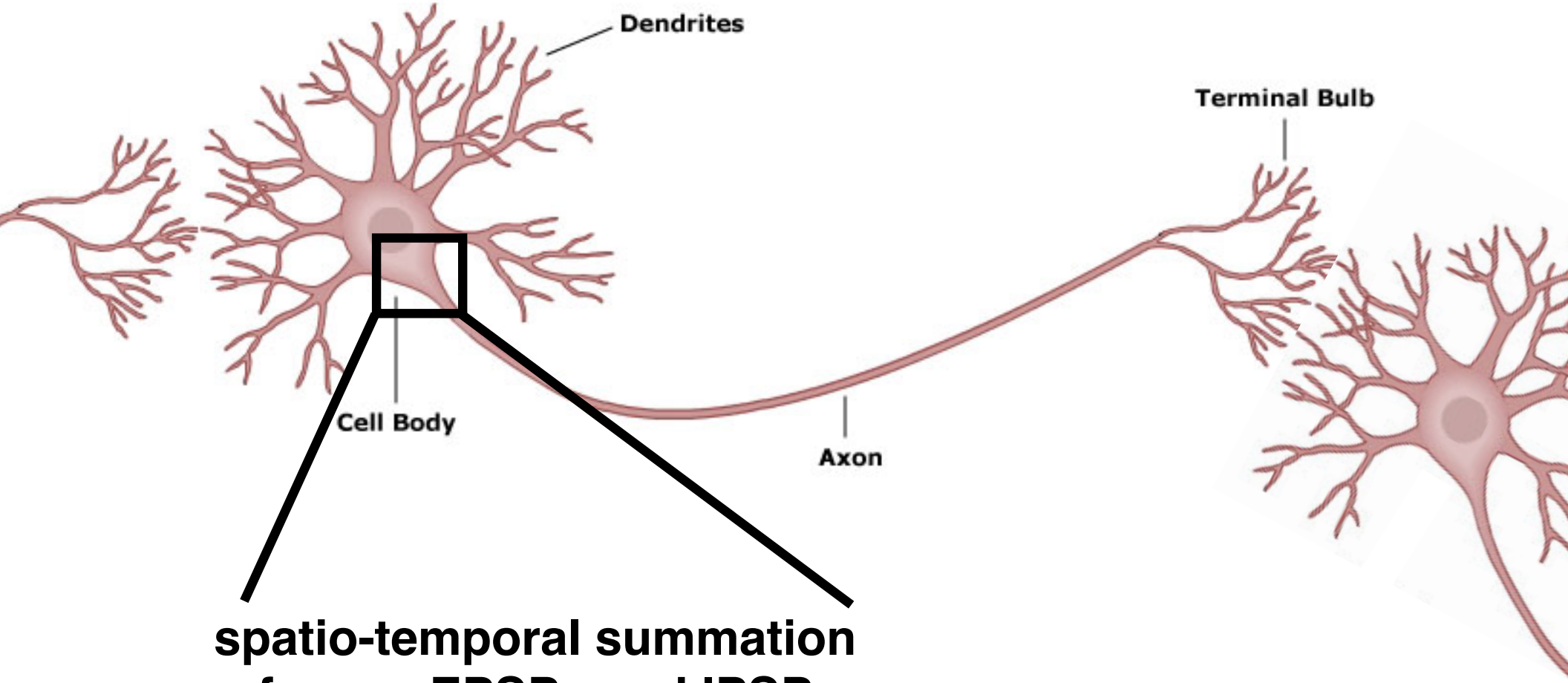


inputs can be positive (+) or negative(-)



EPSPs (excitatory postsynaptic potentials) positive changes in membrane potential caused by excitatory pre-synaptic neurons (e.g., glutamate and NMDA receptors)

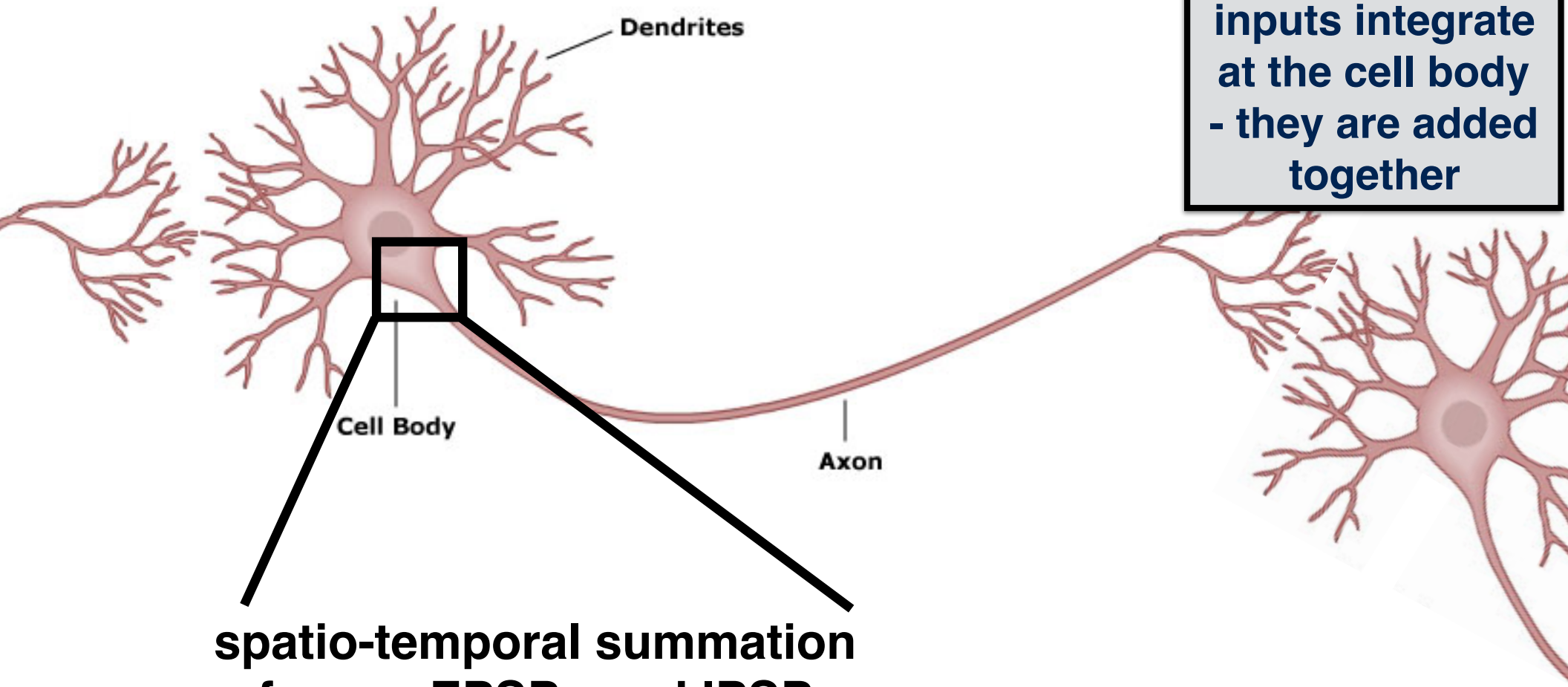
IPSPs (inhibitory postsynaptic potentials) negative changes in membrane potential caused by inhibitory pre-synaptic neurons (e.g., GABA and GABA receptors)



spatio-temporal summation of many EPSPs and IPSPs

from many of synaptic stimulations in close spatio-temporal proximity

potentials spread, but dissipate over space and time because neurons not good conductors

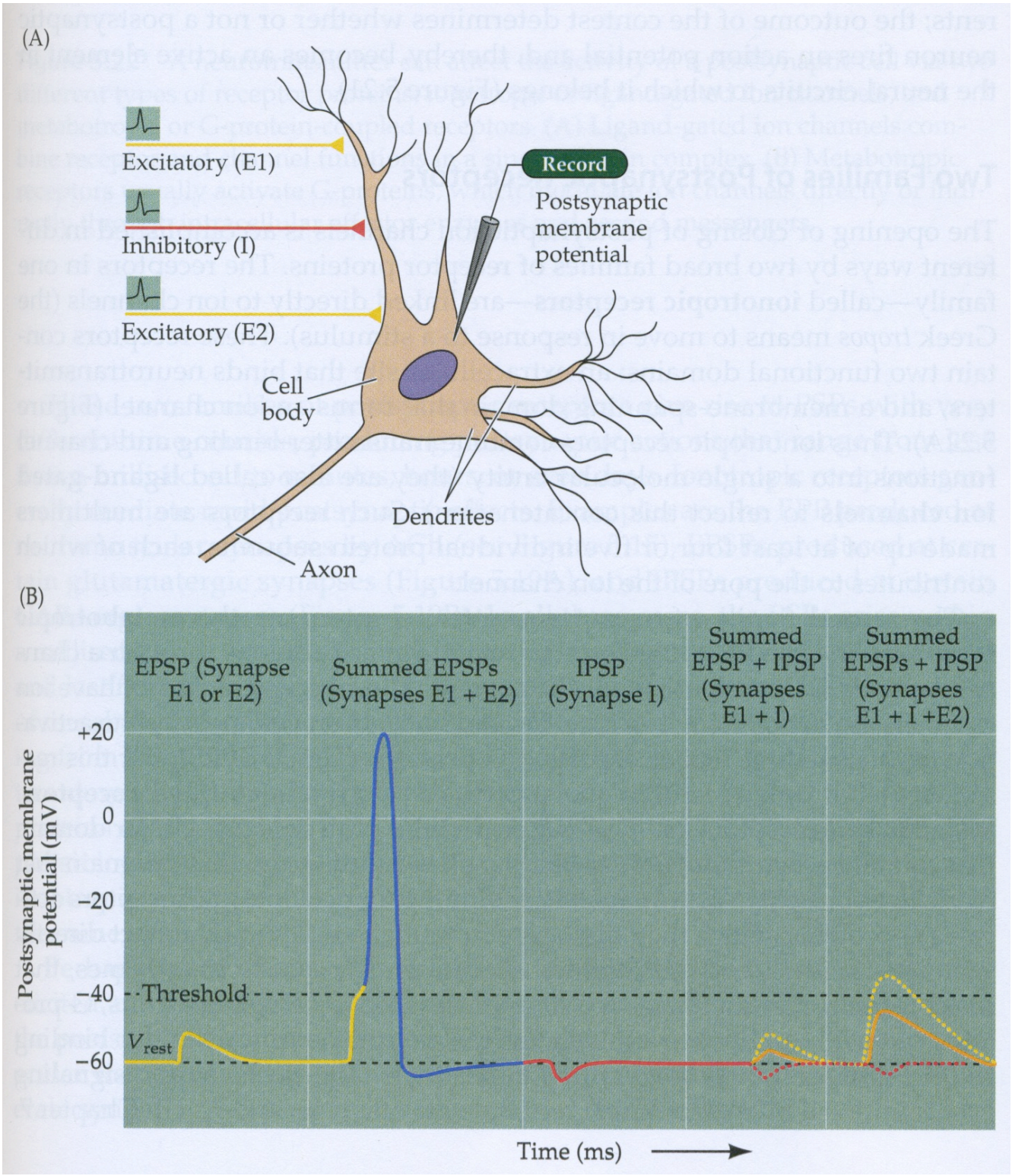


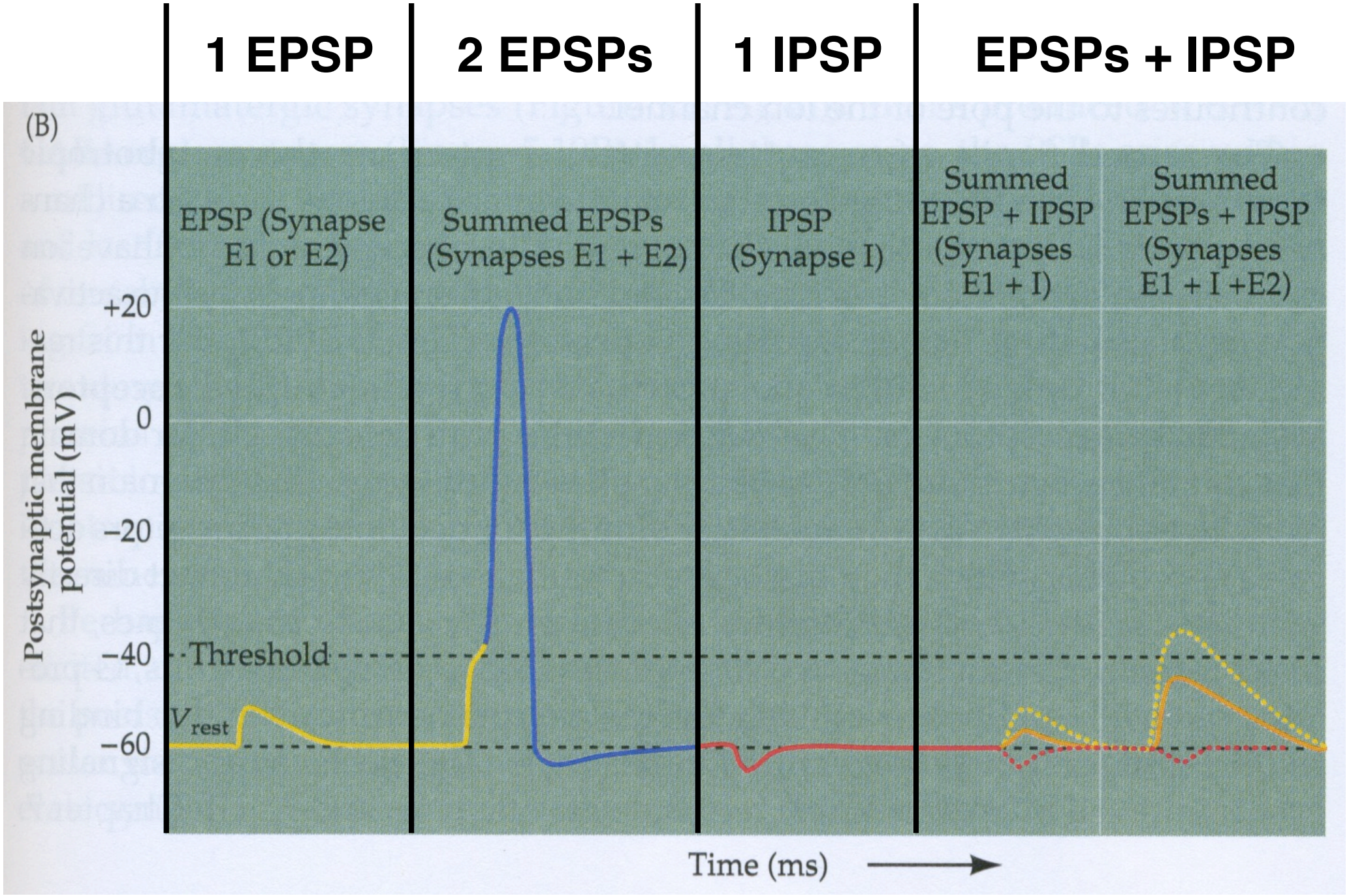
inputs integrate at the cell body - they are added together

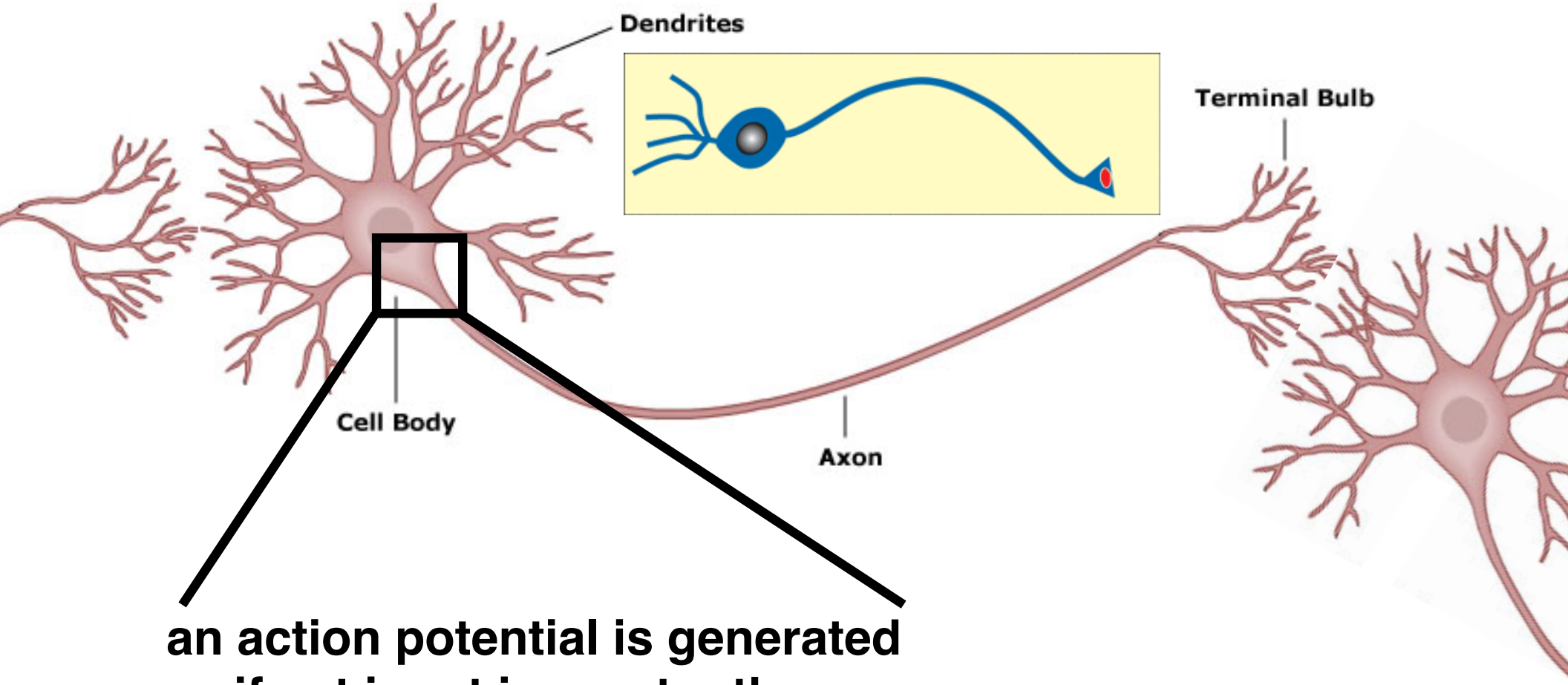
spatio-temporal summation of many EPSPs and IPSPs

from many of synaptic stimulations in close spatio-temporal proximity

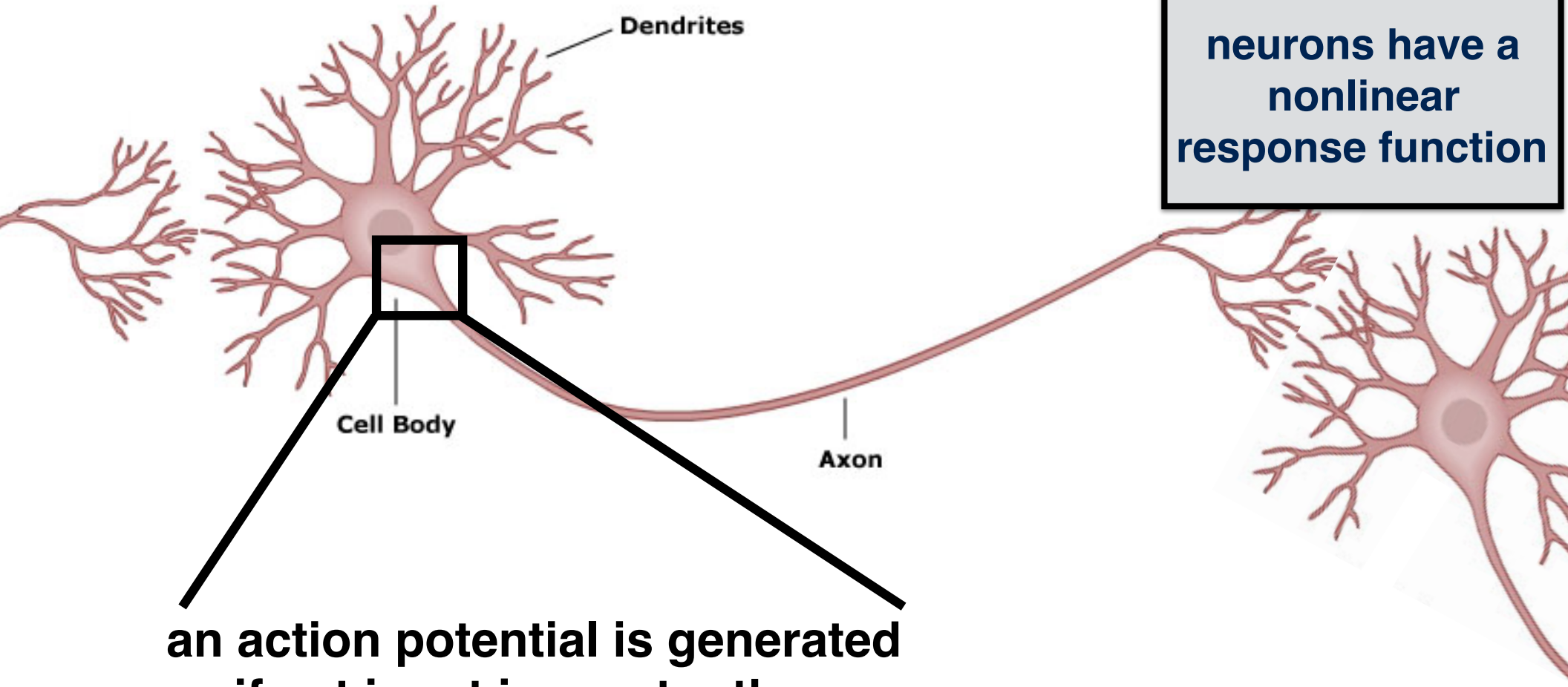
potentials spread, but dissipate over space and time because neurons not good conductors


(A)
Excitatory (E1)
Inhibitory (I)
Excitatory (E2)
Record
Postsynaptic membrane potential
Cell body
Dendrites
Axon
(B)
EPSP (Synapse E1 or E2)
Summed EPSPs (Synapses E1 + E2)
IPSP (Synapse I)
Summed EPSP + IPSP (Synapses E1 + I)
Summed EPSPs + IPSP (Synapses E1 + I +E2)
Postsynaptic membrane potential (mV)
+20
0
−20
−40
−60
Threshold
V_{rest}
Time (ms)


1 EPSP
2 EPSPs
1 IPSP
EPSPs + IPSP
(B)
EPSP (Synapse E1 or E2)
Summed EPSPs (Synapses E1 + E2)
IPSP (Synapse I)
Summed EPSP + IPSP (Synapses E1 + I)
Summed EPSPs + IPSP (Synapses E1 + I +E2)
Postsynaptic membrane potential (mV)
+20
0
−20
−40
−60
Threshold
V_{rest}
Time (ms)



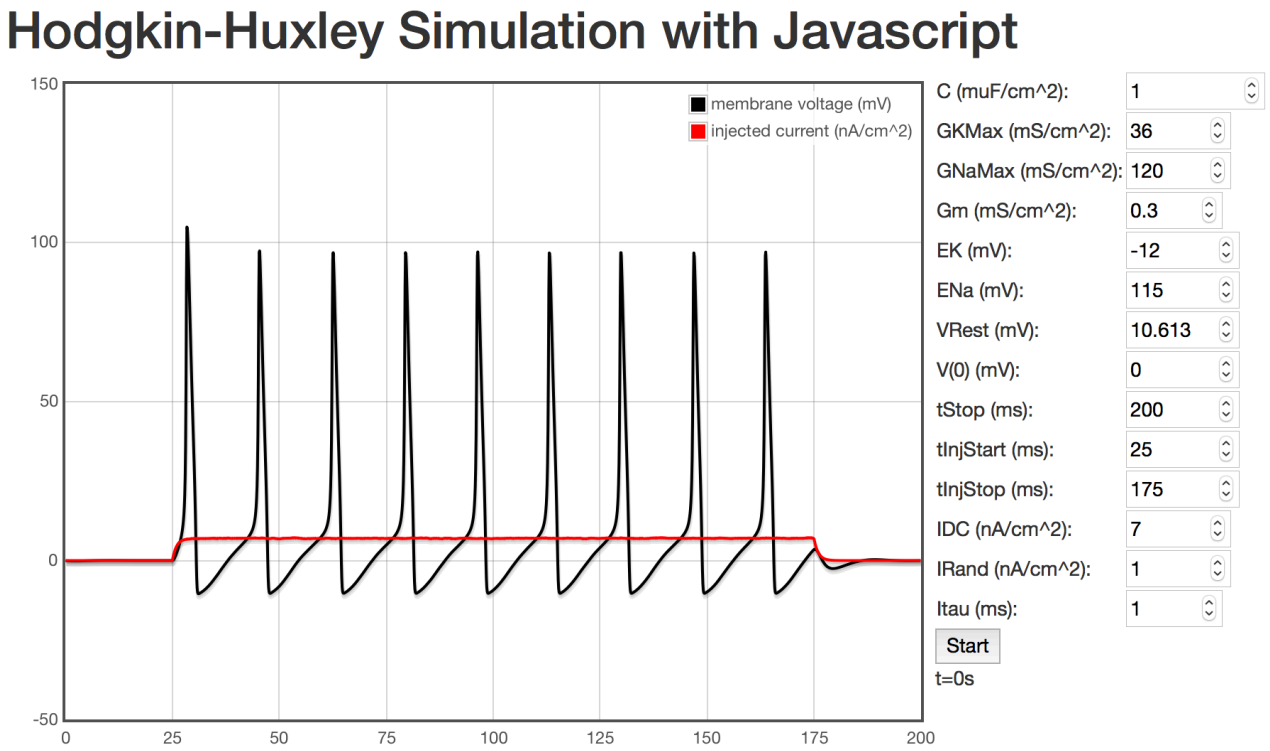
an action potential is generated if net input is greater than an activation threshold



neurons have a nonlinear response function

an action potential is generated if net input is greater than an activation threshold

action potential simulator

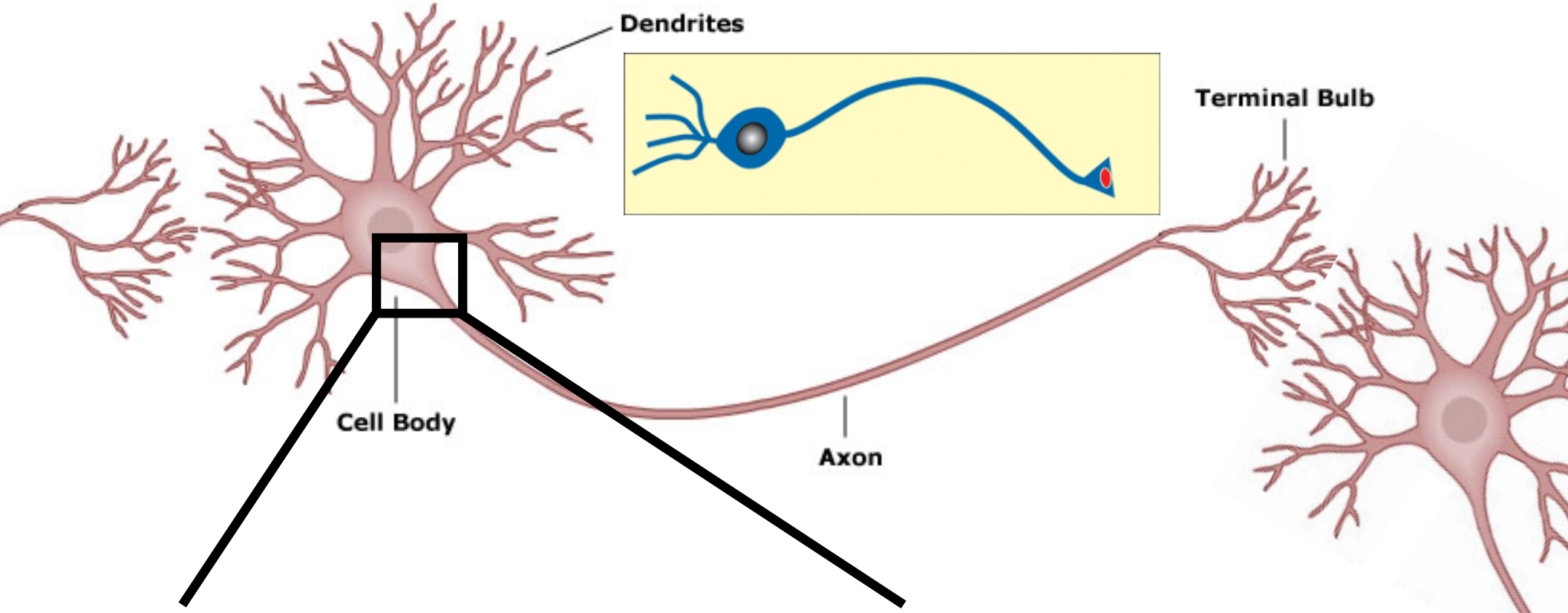


http://myselph.de/hodgkinHuxley.html

IDC (input/injected current)

IRand (random variability in current)

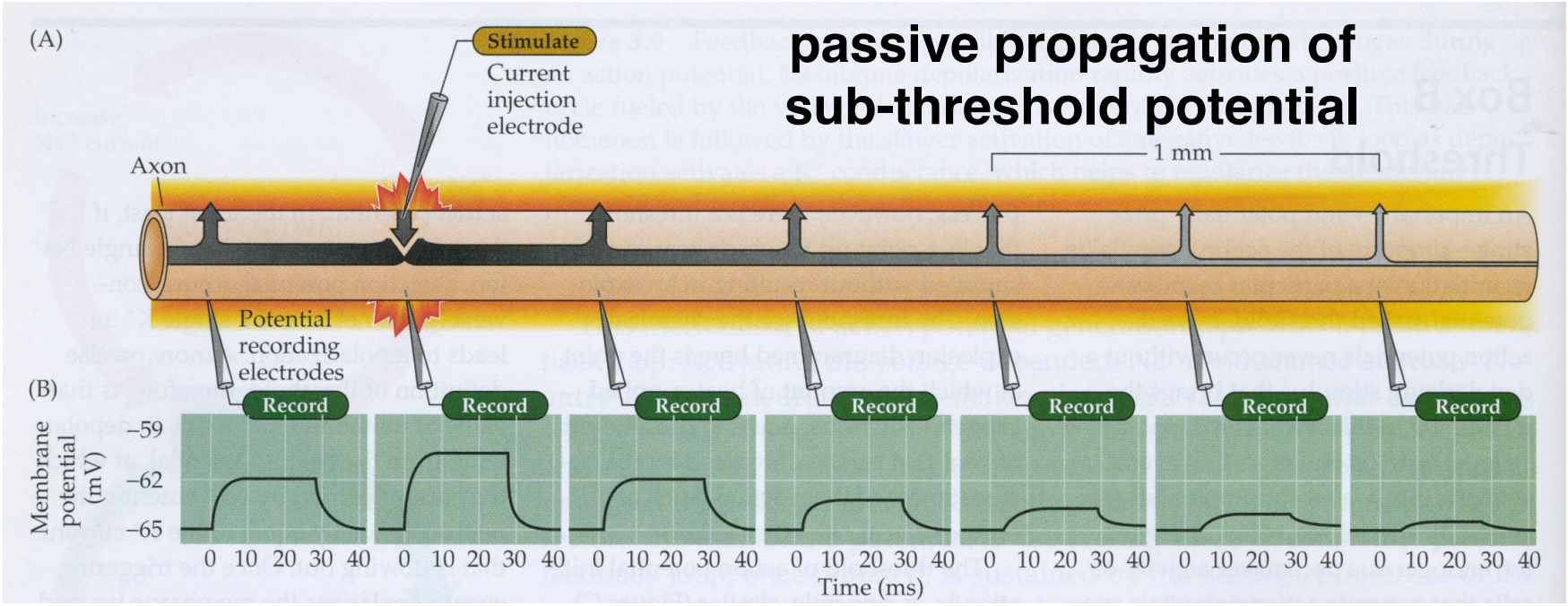
try IDC=2.355 vs. IDC=2.356 with IRand=0

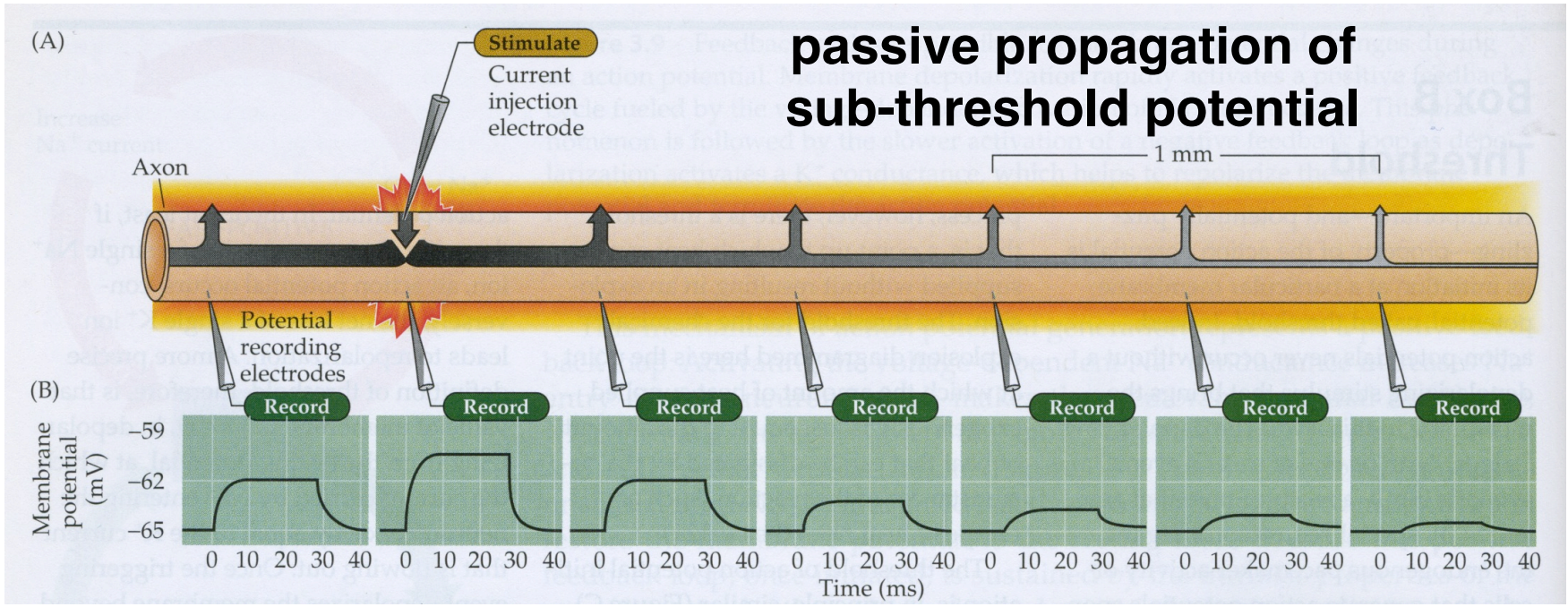


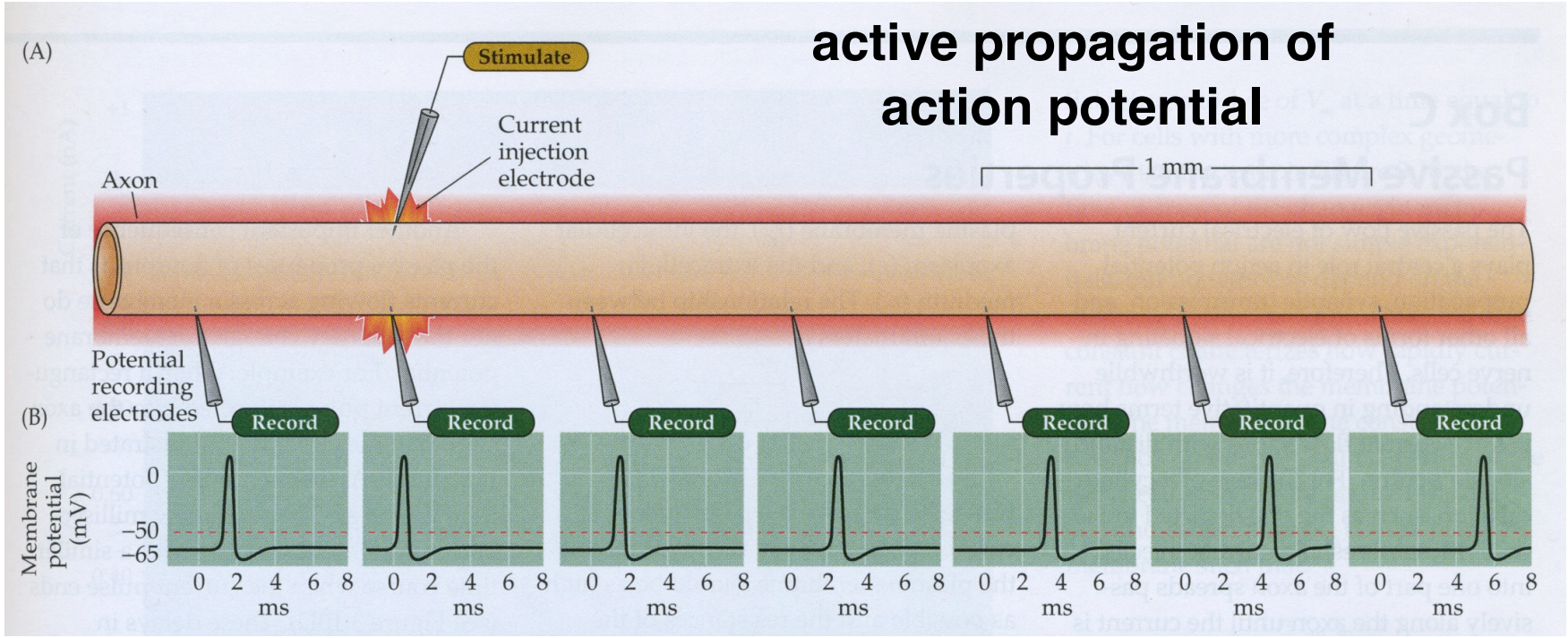
an action potential is generated if net input is greater than an activation threshold

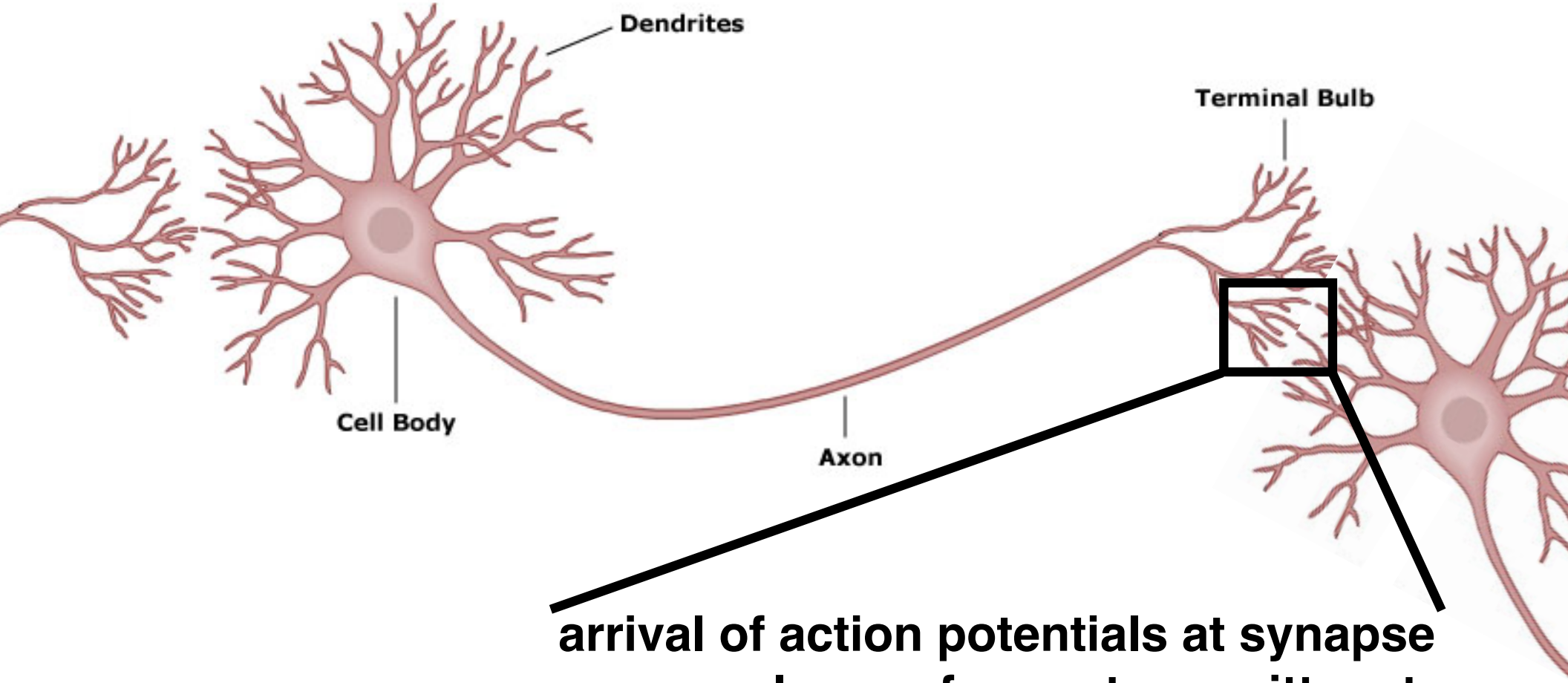
action potential is nature's way of making the axon conductive over relatively long distances

passive propagation of sub-threshold potential




passive propagation of
sub-threshold potential
(A)
Stimulate
Current
injection
electrode
Axon
1 mm
Potential
recording
electrodes
(B)
Record
Membrane
potential
(mV)
–59
–62
–65
0 10 20 30 40
Time (ms)

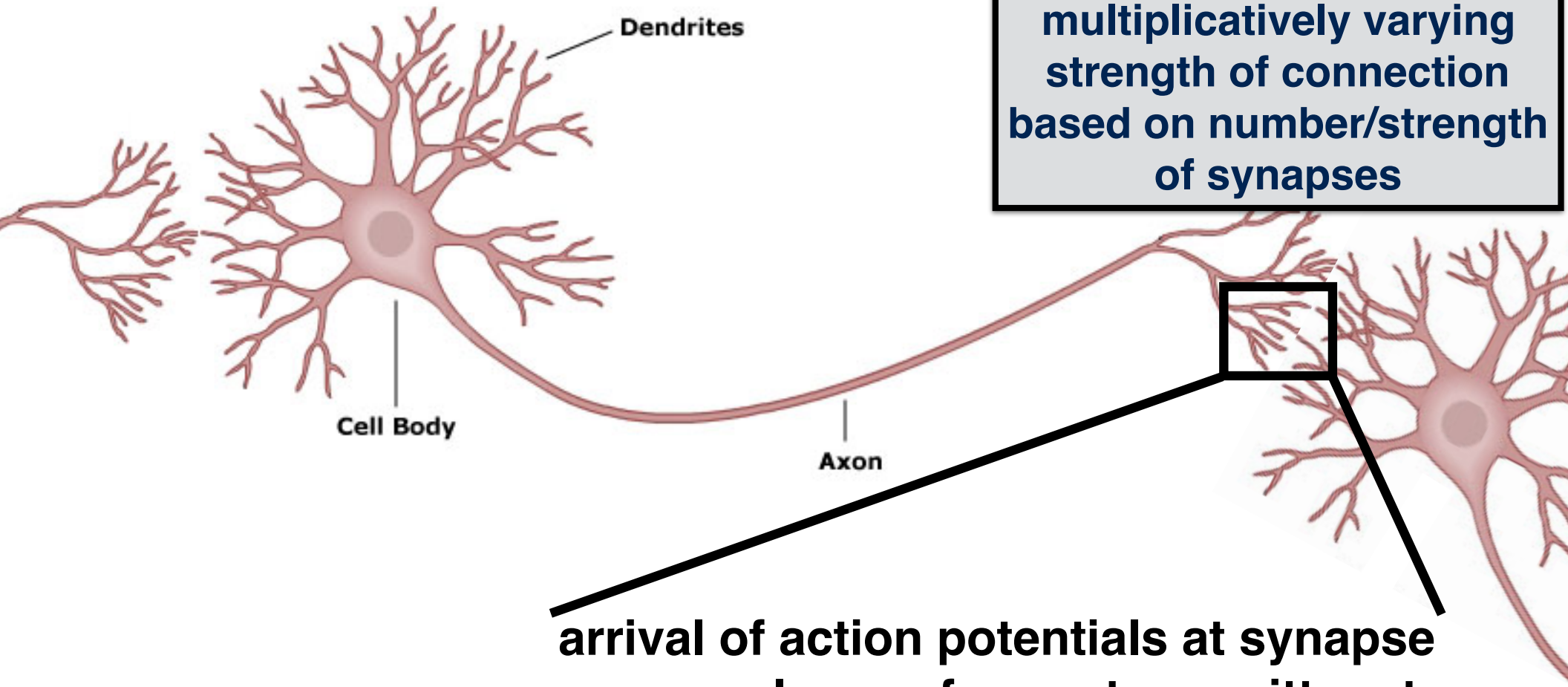

active propagation of
action potential
(A)
Stimulate
Current
injection
electrode
Axon
1 mm
Potential
recording
electrodes
(B)
Record
Membrane
potential
(mV)
0
–50
–65
0 2 4 6 8
ms



arrival of action potentials at synapse causes release of neurotransmitters to post-synaptic neuron

more and/or stronger synapses mean pre-synaptic neuron has more influence over post-synaptic neuron

synapses <u>multiply</u> effect of action potentials



multiplicatively varying strength of connection based on number/strength of synapses

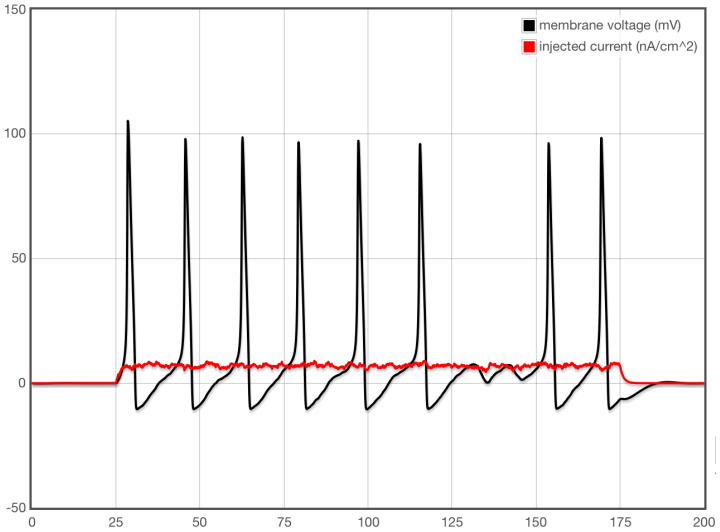
arrival of action potentials at synapse causes release of neurotransmitters to post-synaptic neuron

more and/or stronger synapses mean pre-synaptic neuron has more influence over post-synaptic neuron

synapses multiply effect of action potentials

Levels of Neuron Modeling

- detailed model of action potentials (Hodgkin-Huxley model)



$$\frac{dV}{dt} = \frac{1}{C}(-I_{NA} - I_K - I_{leak} - I_{input})$$

$$\frac{dV}{dt} = \frac{1}{C}\left(-g_{NA}m^3h(V - E_{NA}) - g_K n^4 (V - E_K) - g_{leak}(V - E_{leak}) + I_{input}\right)$$

$$\frac{dm}{dt} = \frac{1}{\tau_m(V)}(-m + M(V))$$

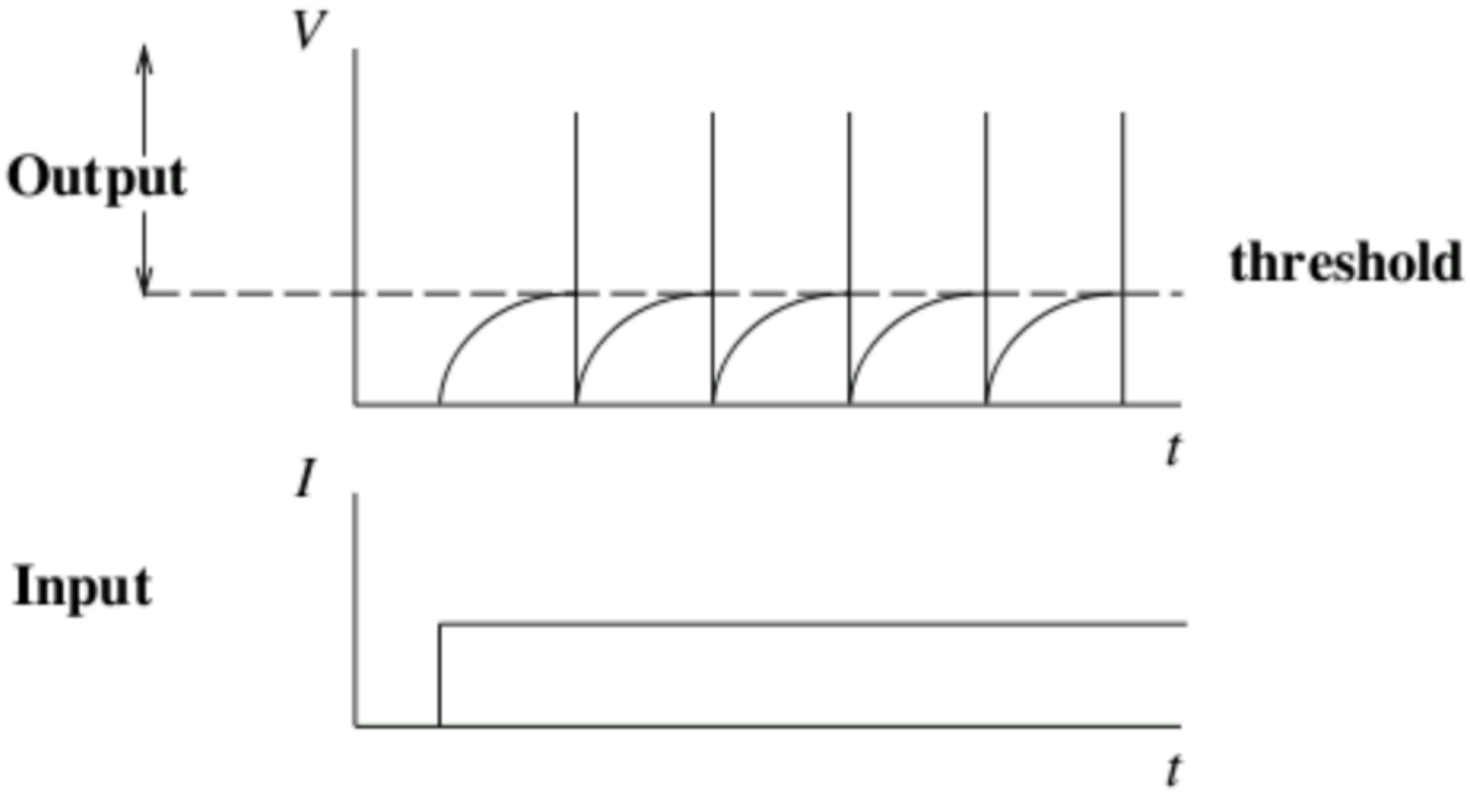
$$\frac{dh}{dt} = \frac{1}{\tau_h(V)}(-h + H(V))$$

$$\frac{dn}{dt} = \frac{1}{\tau_n(V)}(-n + N(V))$$

Levels of Neuron Modeling

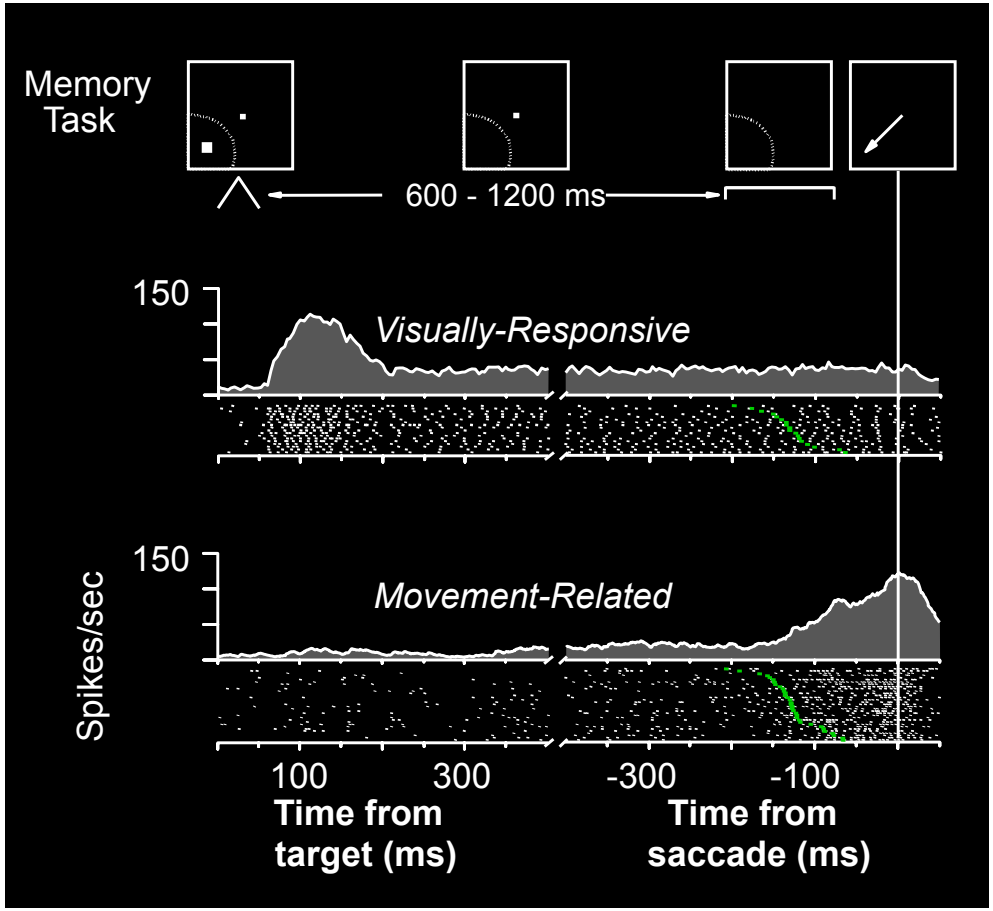
- detailed model of action potentials (Hodgkin-Huxley model)

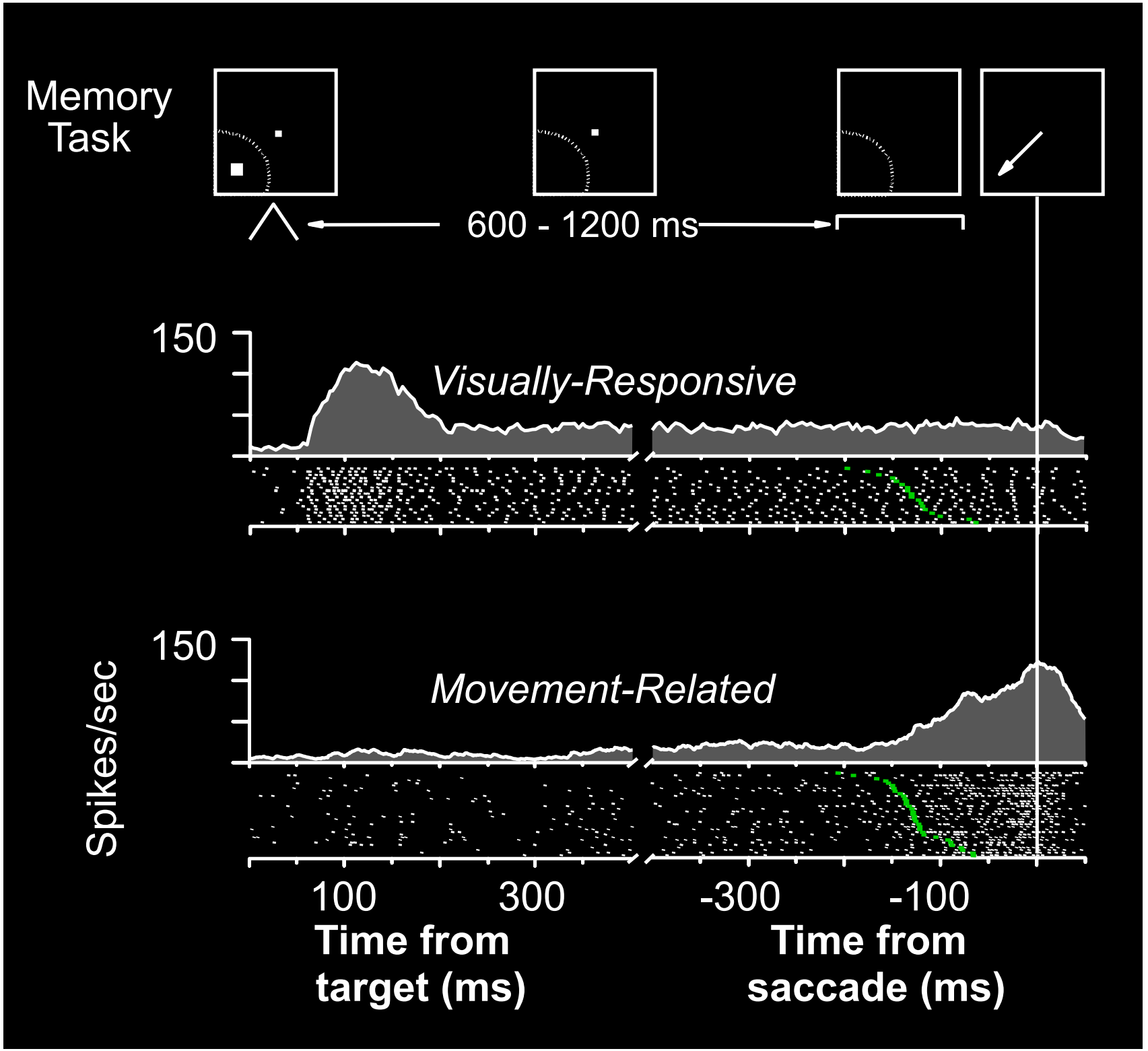
- integrate-and-fire (spiking) neuron



Levels of Neuron Modeling

- detailed model of action potentials (Hodgkin-Huxley model)
- integrate-and-fire (spiking) neuron
- rate-coded neuron




Memory
Task
600 - 1200 ms
150
Visually-Responsive
150
Movement-Related
Spikes/sec
100
300
-300
-100
Time from
target (ms)
Time from
saccade (ms)

Levels of Neuron Modeling

- detailed model of action potentials (Hodgkin-Huxley model)

- integrate-and-fire (spiking) neuron

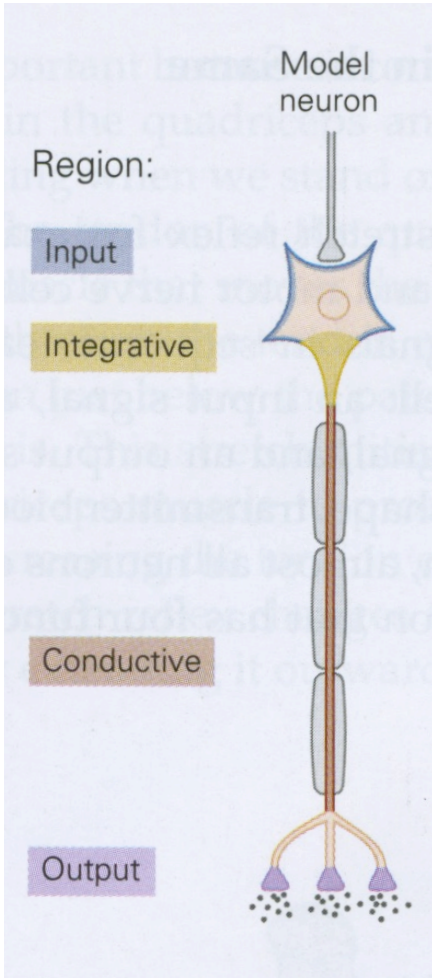
- rate-coded neuron

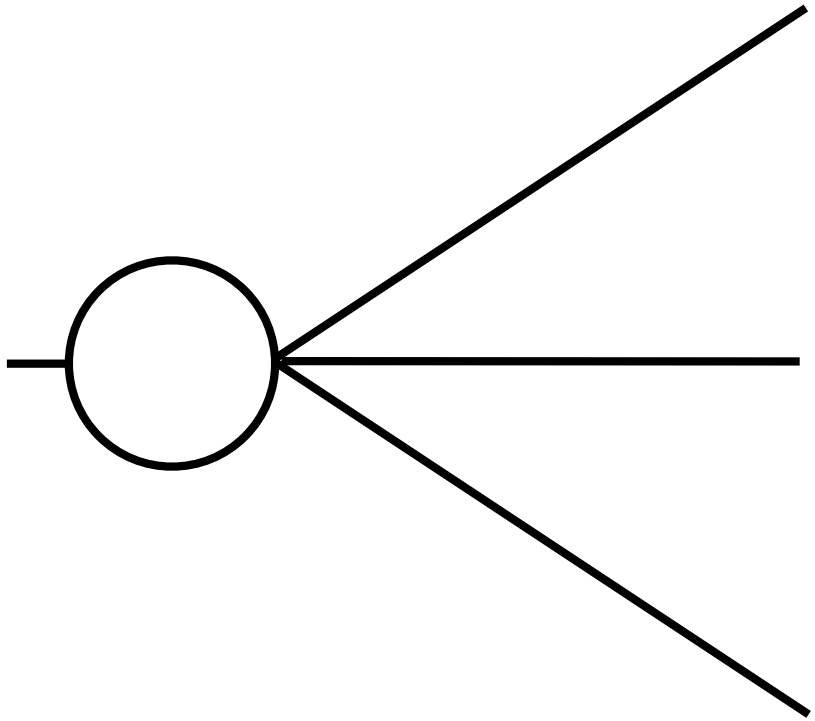
  we'll start by considering neural network models that not only ignore spikes, but largely ignore time altogether

  then consider models that learn, networks that change over time with experience

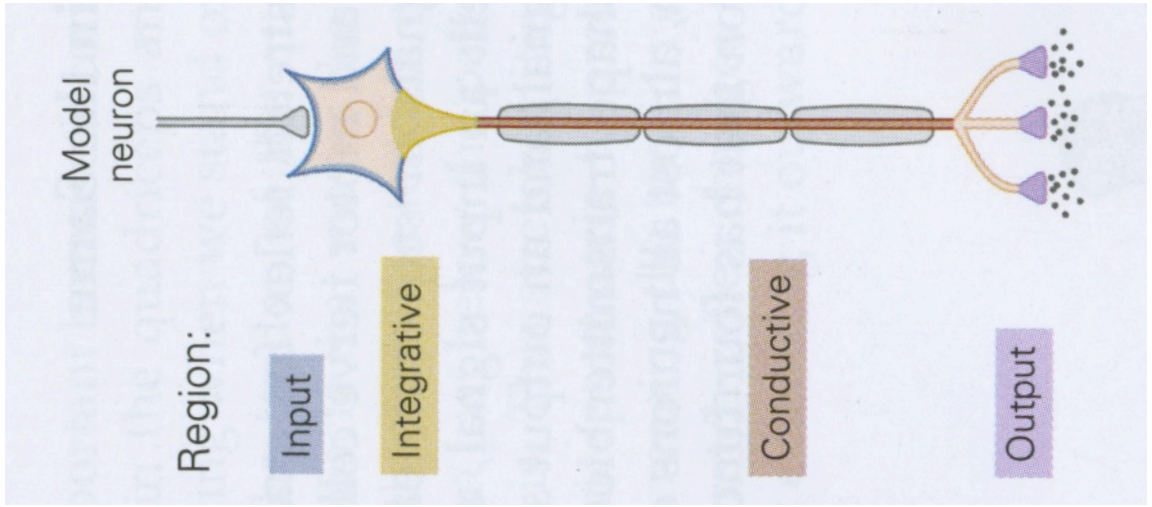
  may consider models that allow for dynamics (recurrent networks, dynamical networks, spiking)

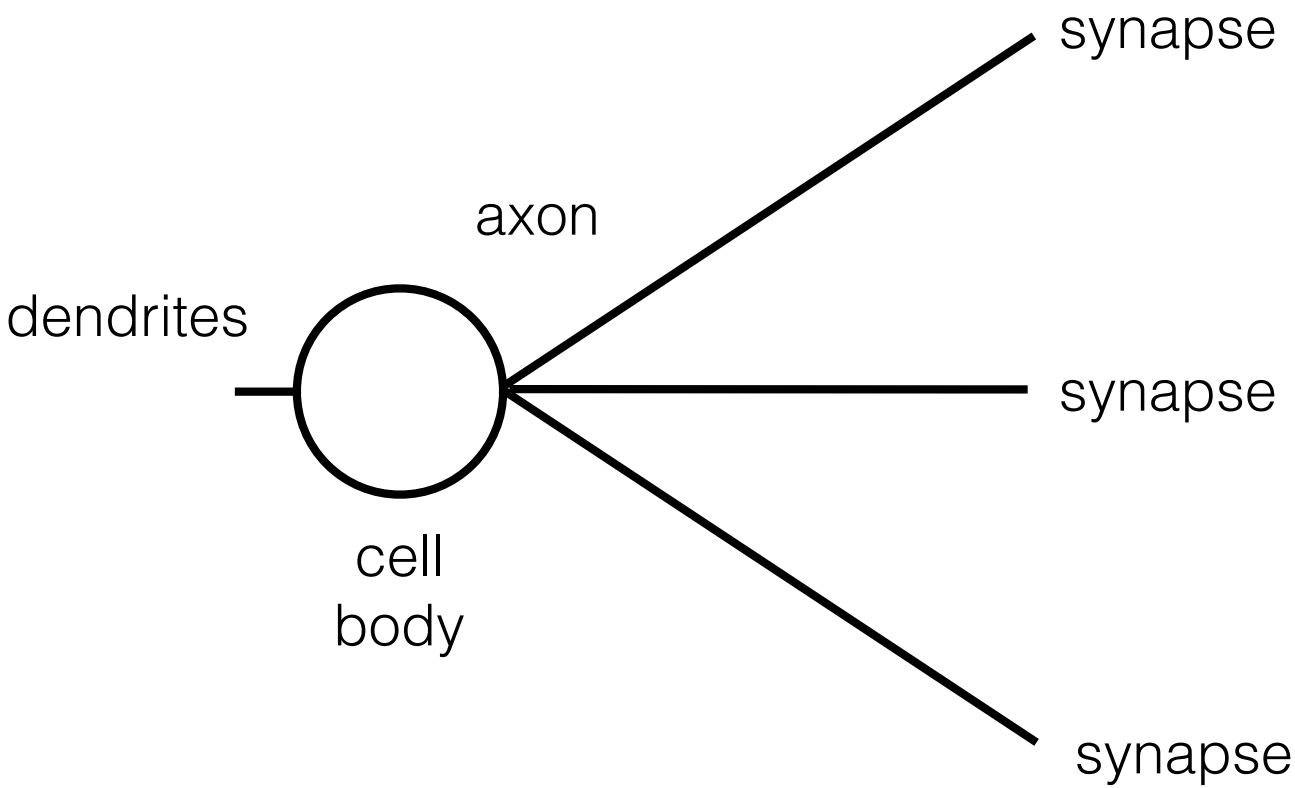
modeling an idealized neuron



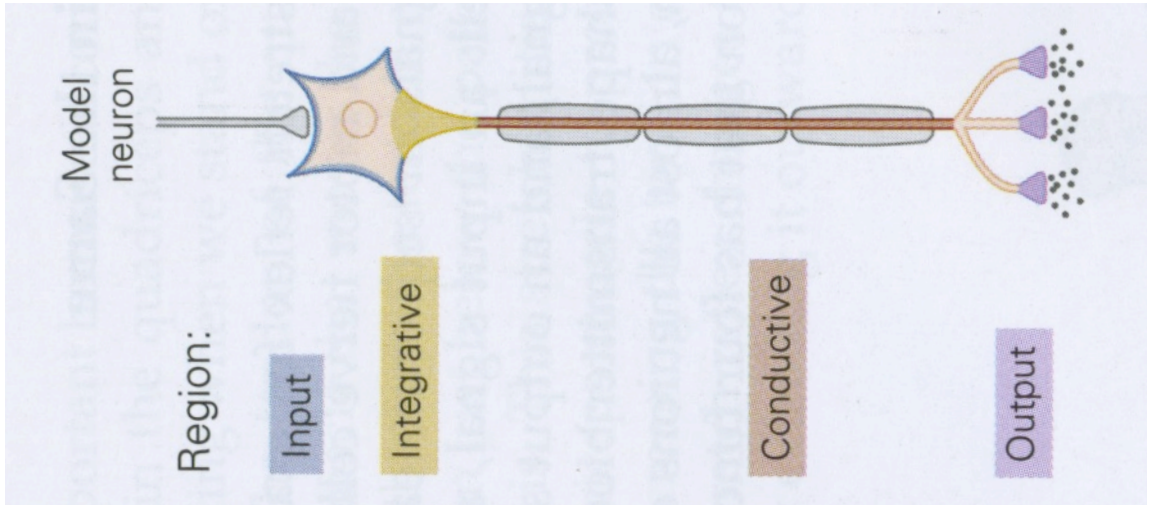


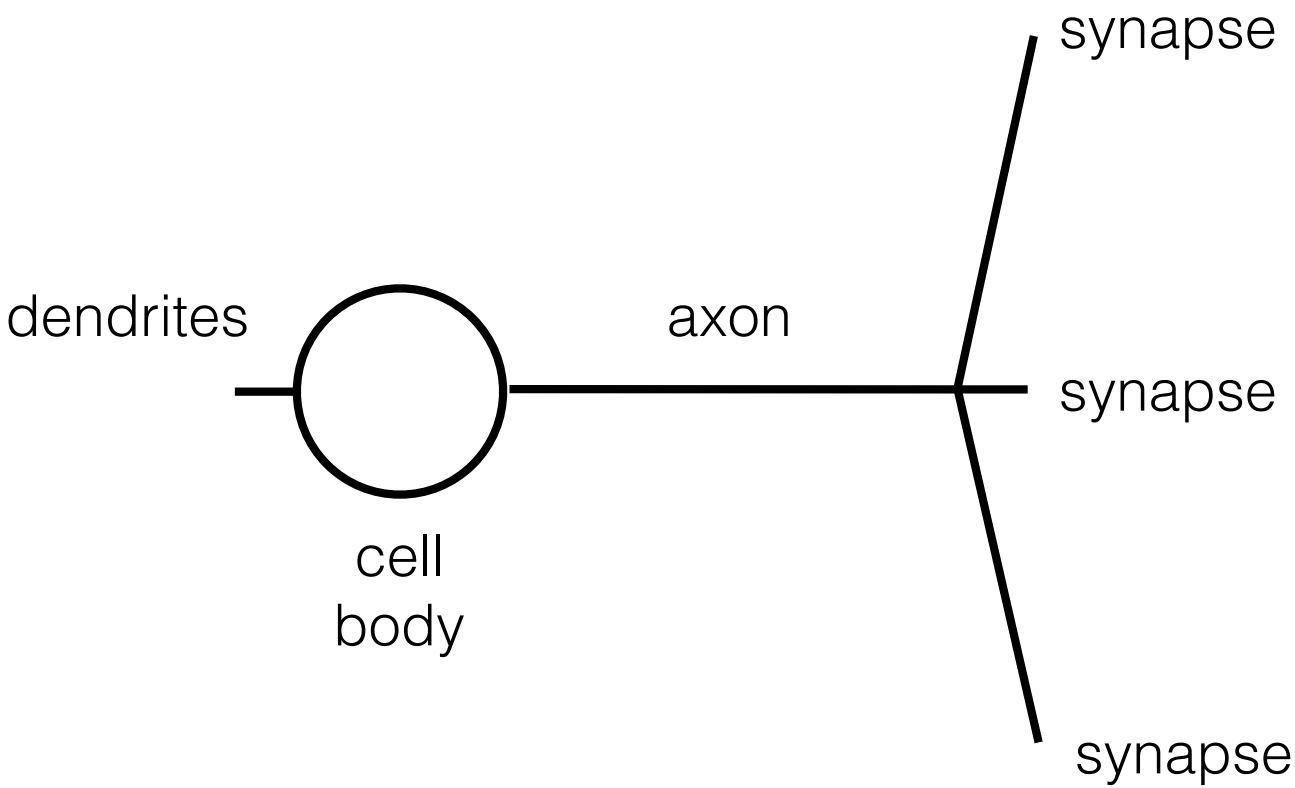
Idealized Neuron



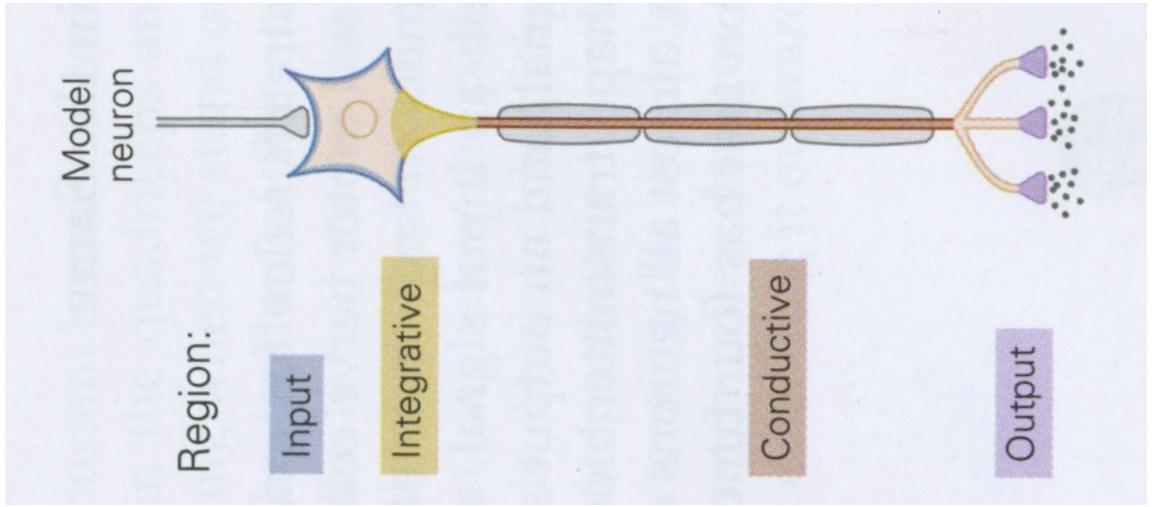


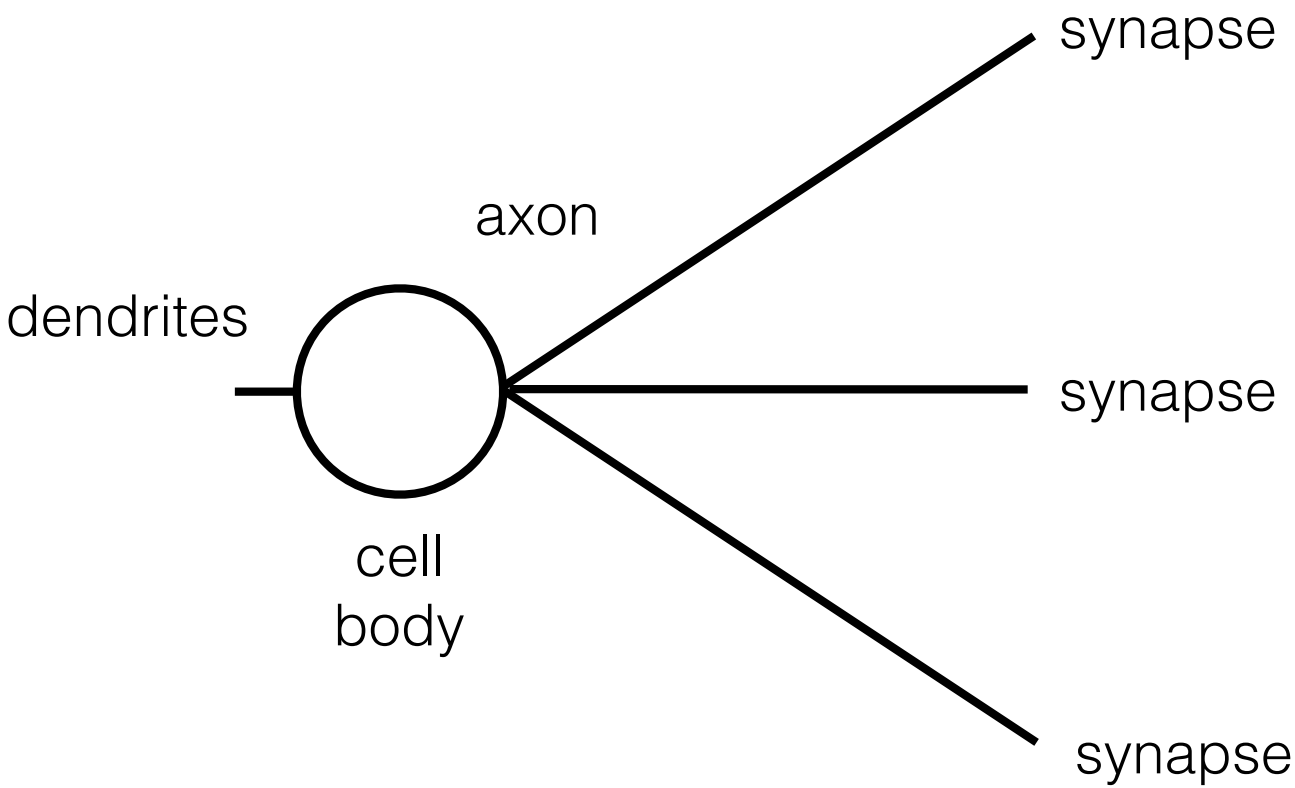
Idealized Neuron





Idealized Neuron



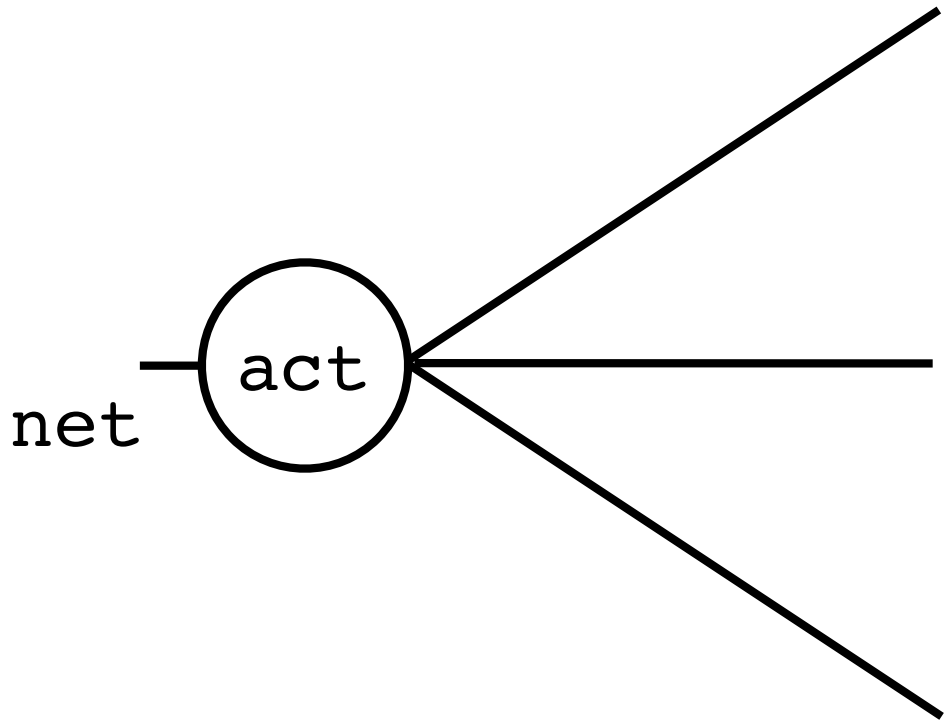


Idealized Neuron

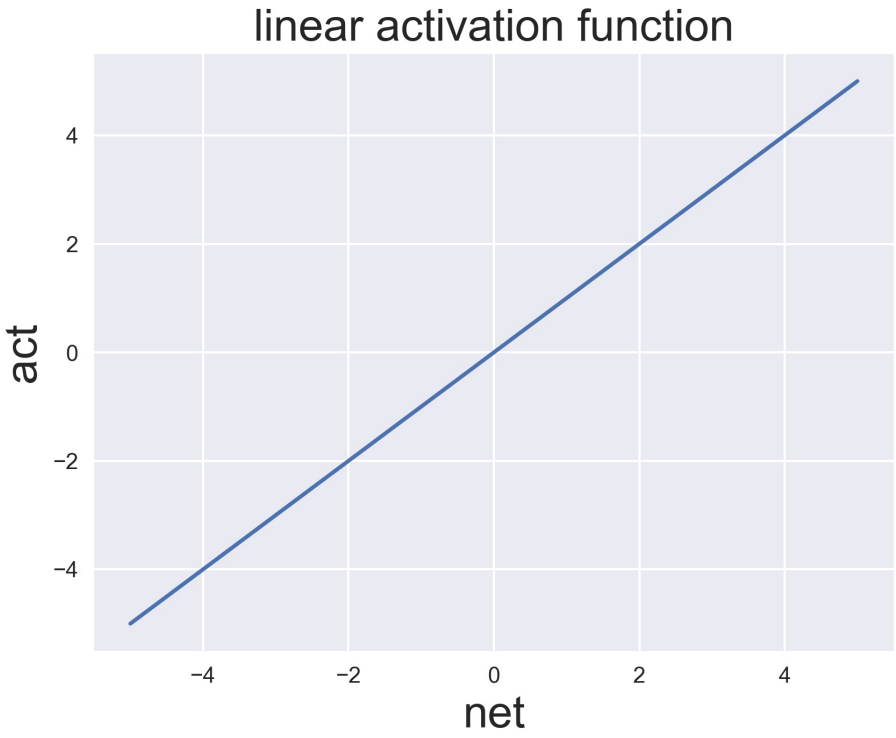
"unit" or "node"

how is the activation of a neuron related to its net input?

where "activation" is akin to spike rate



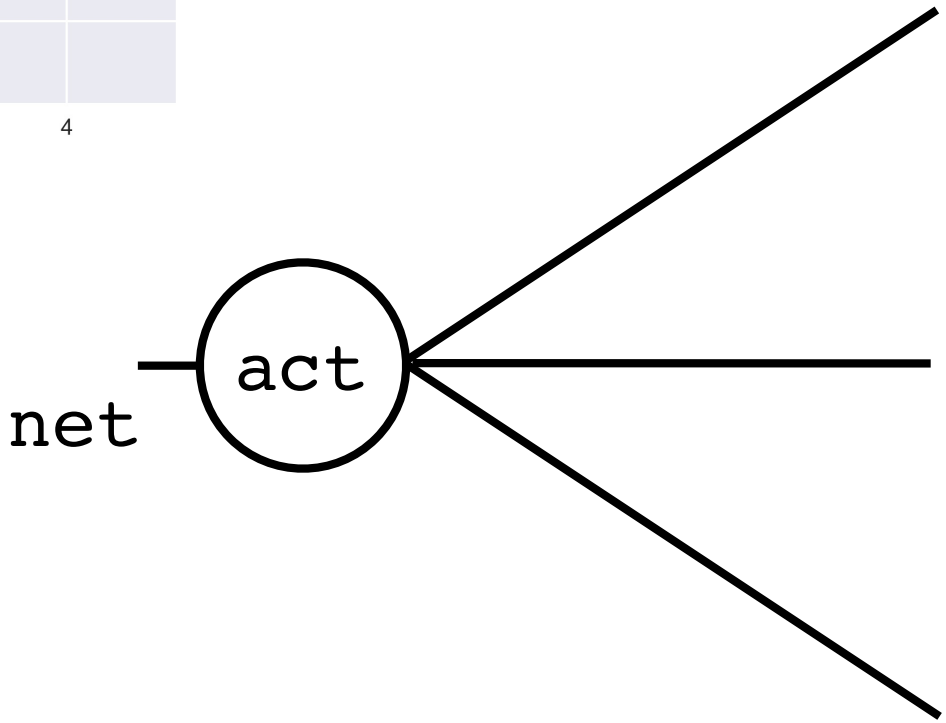
Idealized Neuron



or course this is not biologically plausible if activation is equated with rate

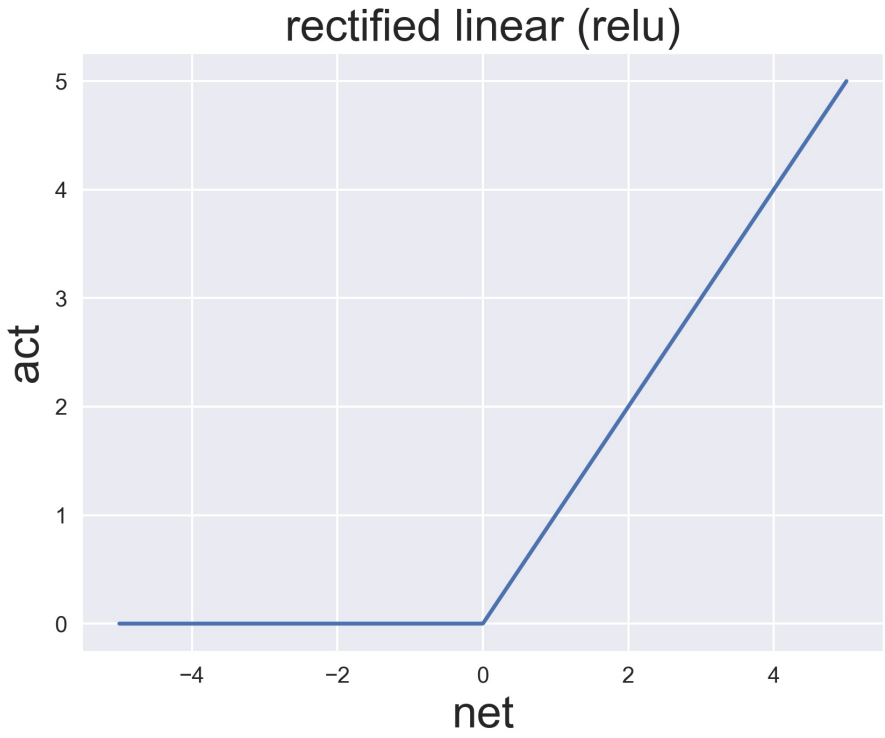
but a unit could instead represent a local circuit or neural population

$$a_j = n_j$$



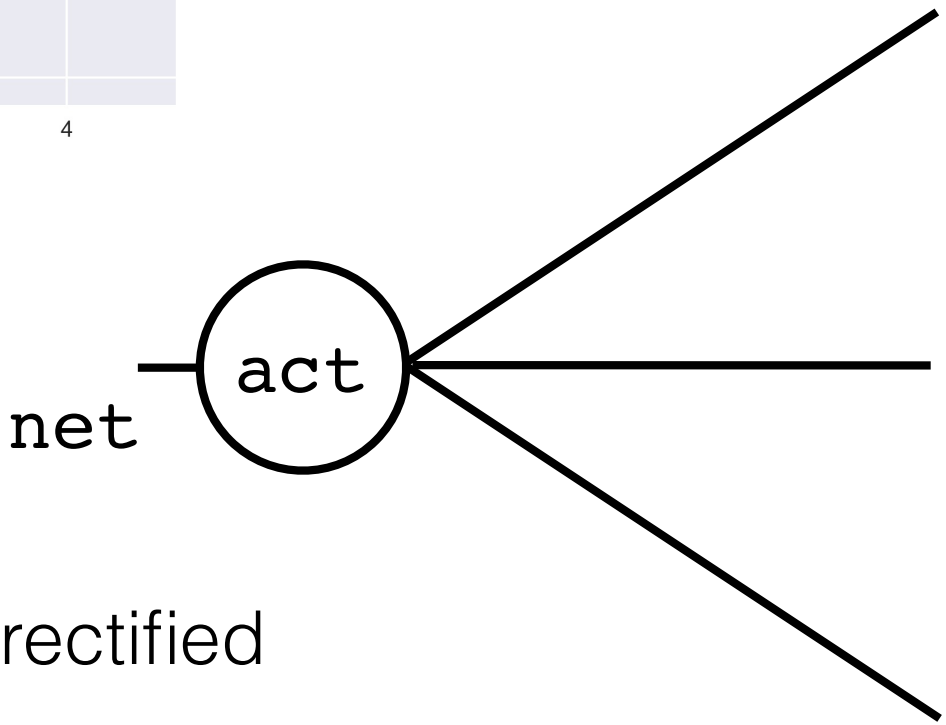
`linear()`

Idealized Neuron



neurons have a nonlinear response function

$$a_j = \begin{cases} 0 & \text{if } n_j \leq 0 \\ n_j & \text{otherwise} \end{cases}$$

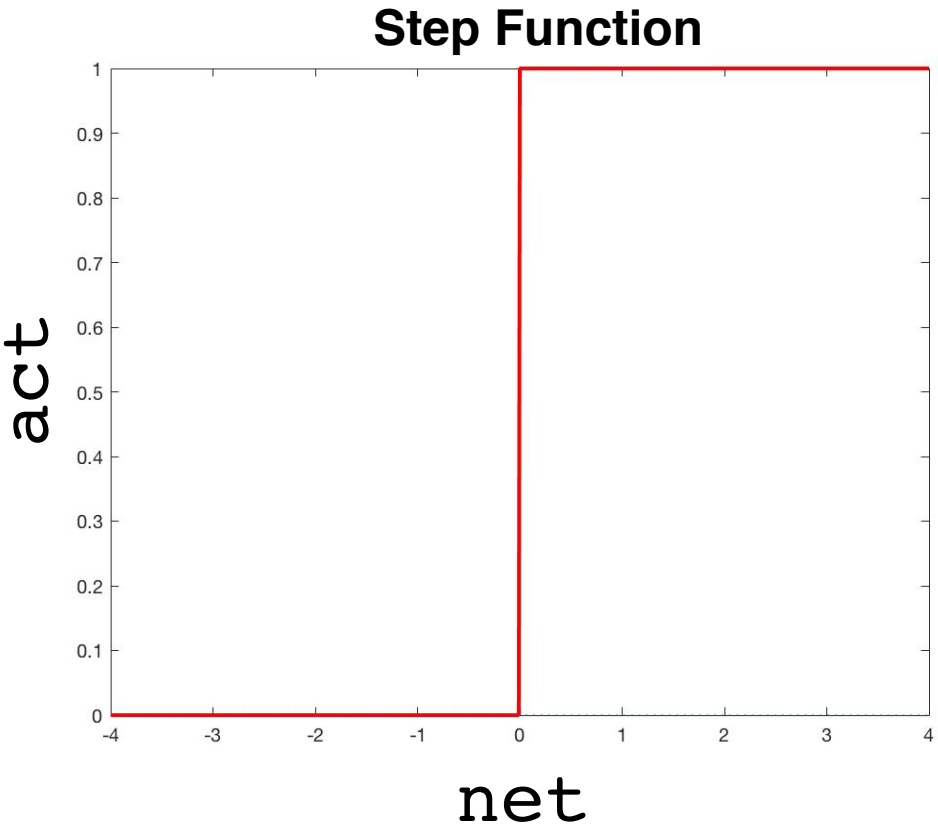


sometimes called a rectified linear unit (or relu)

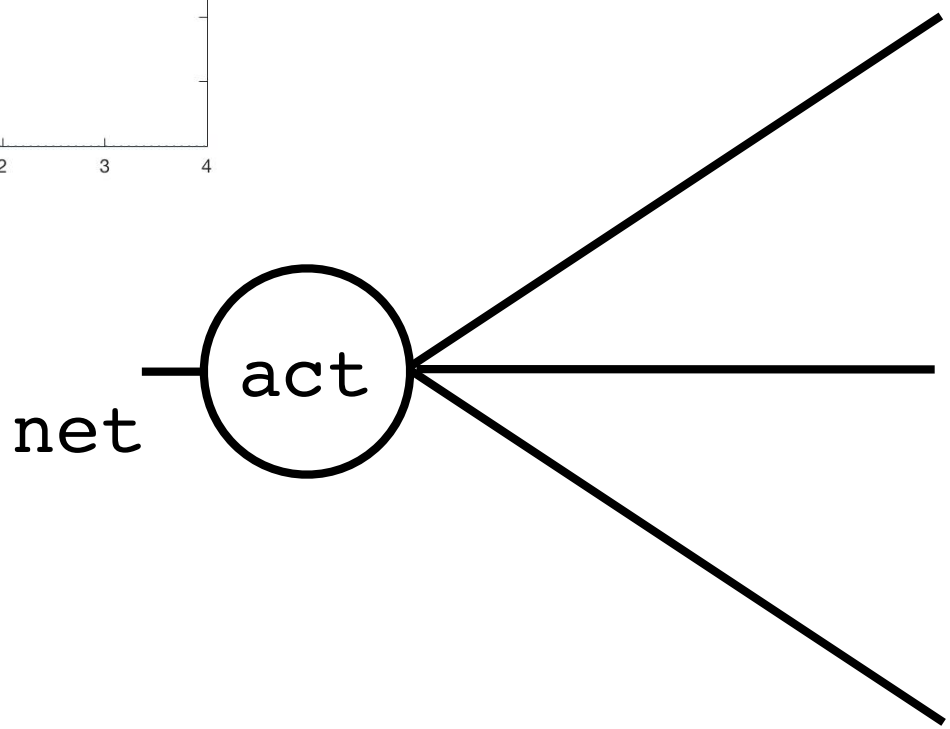
`relu()`

Idealized Neuron

neurons have a nonlinear response function



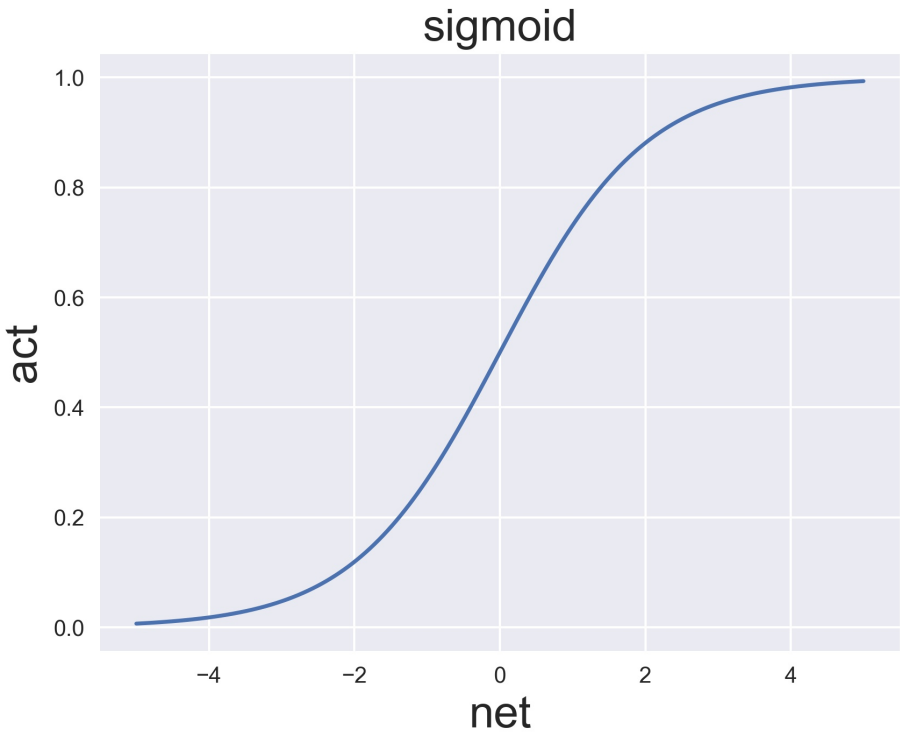
$$a_j = \begin{cases} 0 & \text{if } n_j \le 0 \\ 1 & \text{otherwise} \end{cases}$$



*derivative is zero or undefined

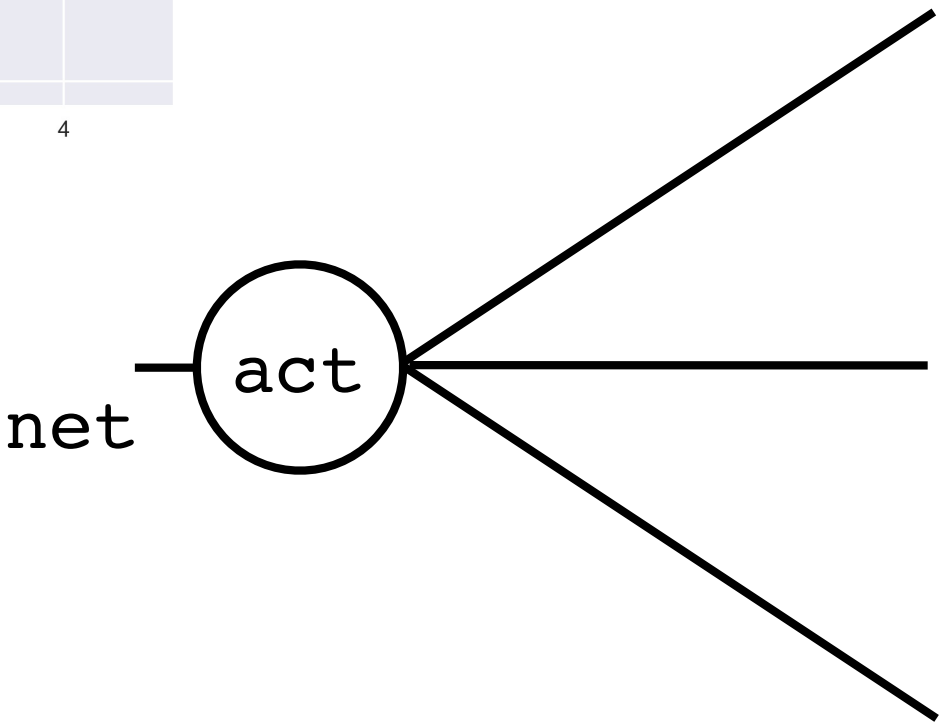
not defined*

Idealized Neuron



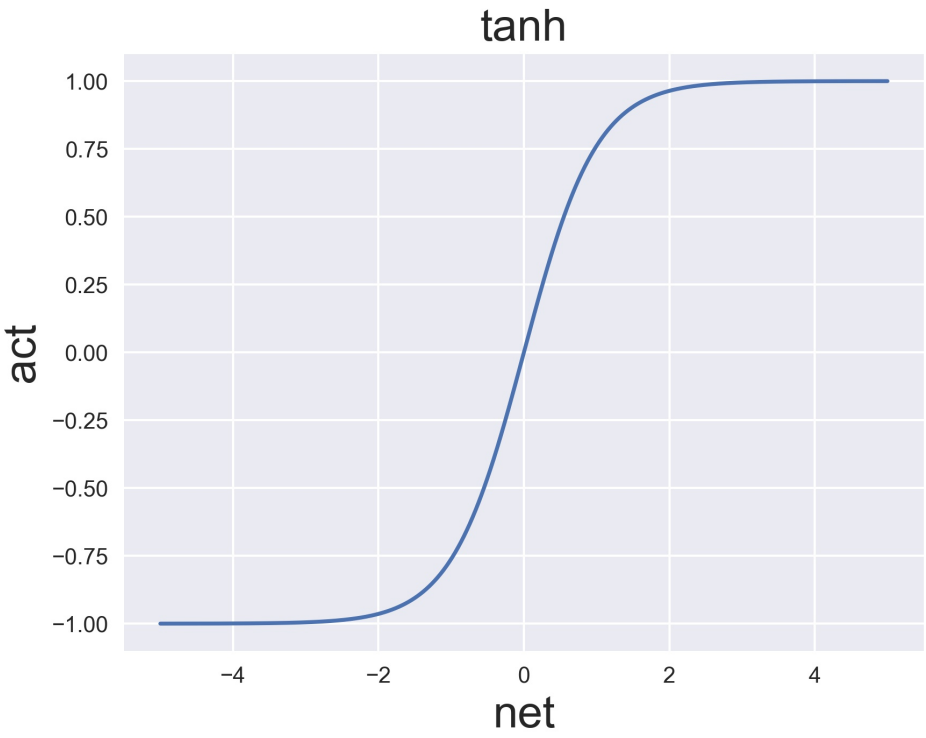
neurons have a nonlinear response function

$$a_j = \frac{1}{1+\exp(-n_j)}$$



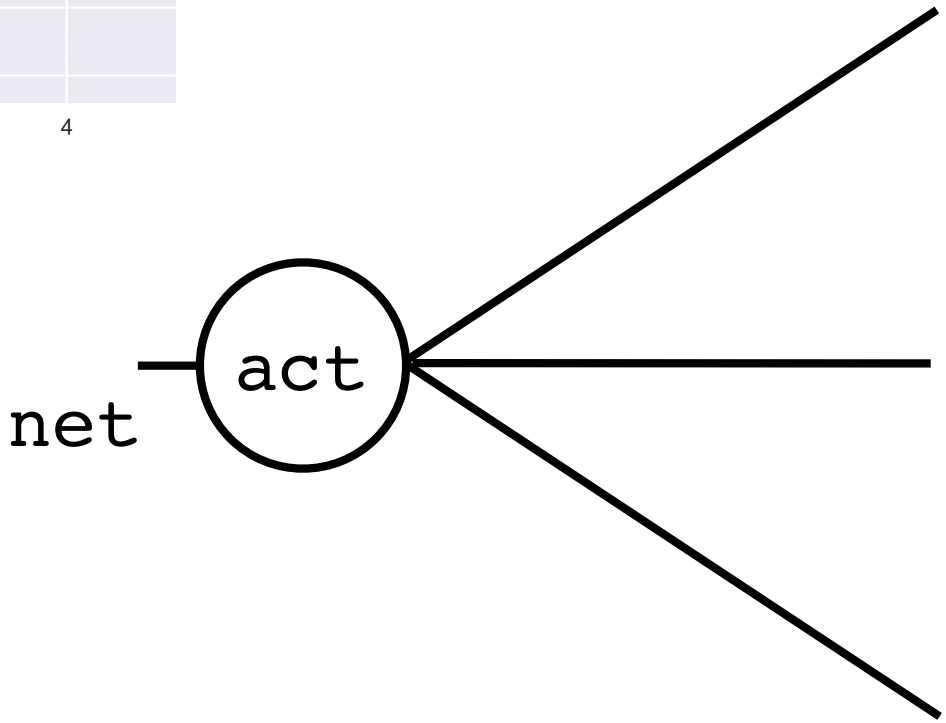
`sigmoid()`

Idealized Neuron



neurons have a nonlinear response function

$$a_j = \frac{\exp(n_j) - \exp(-n_j)}{\exp(n_j) + \exp(-n_j)}$$



`tanh()`

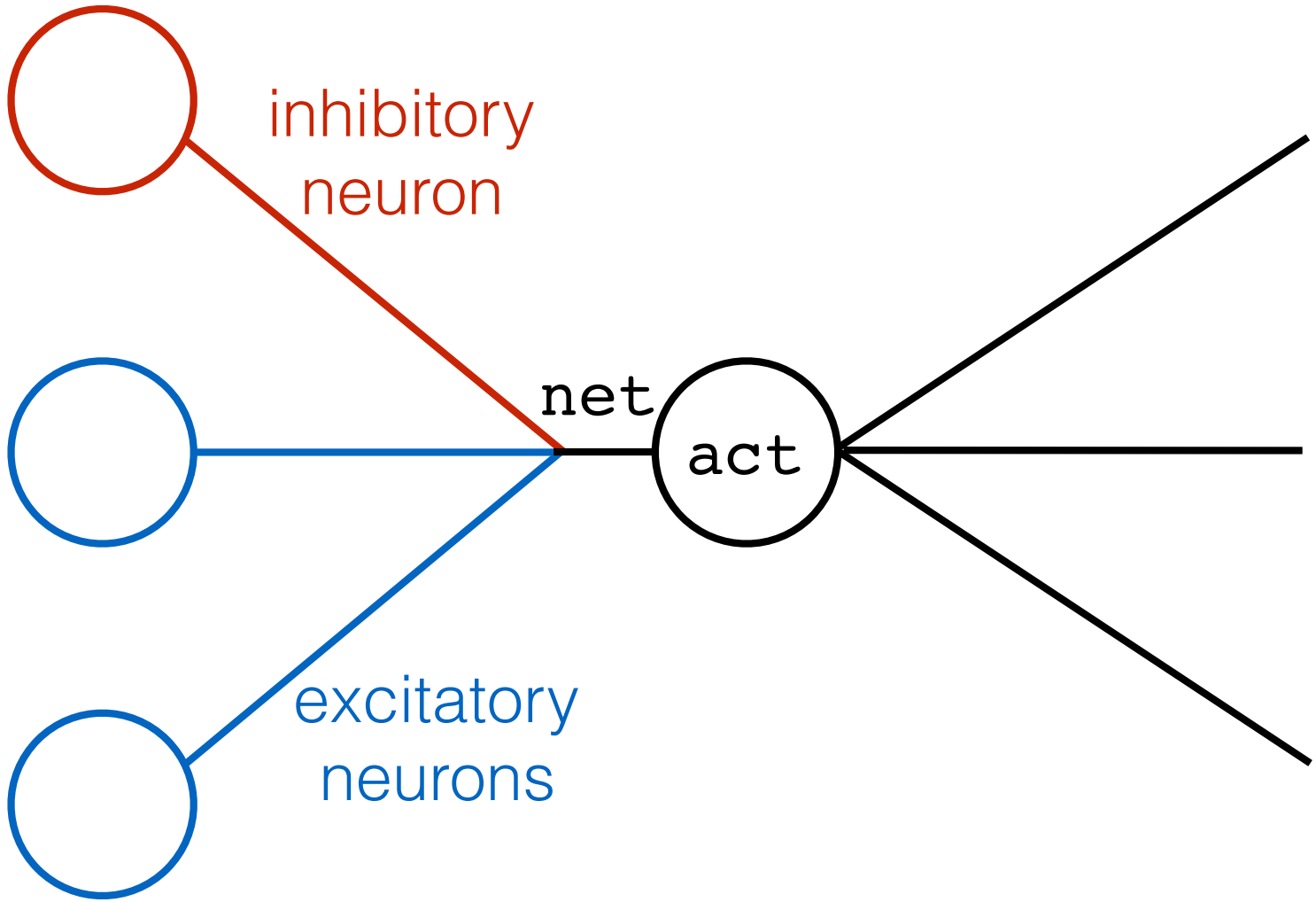
Idealized Neuron

```
import numpy as np
import keras.activations as kact
import tensorflow as tf

session = tf.InteractiveSession()

netnp = np.linspace(-5.0, 5.0, 1000)
nettf = tf.convert_to_tensor(netnp)

acttf = kact.sigmoid(nettf)
actnp = acttf.eval()

session.close()
```

Keras / TensorFlow

see ActivFuncs.py and ActivFuncs.ipynb on Brightspace

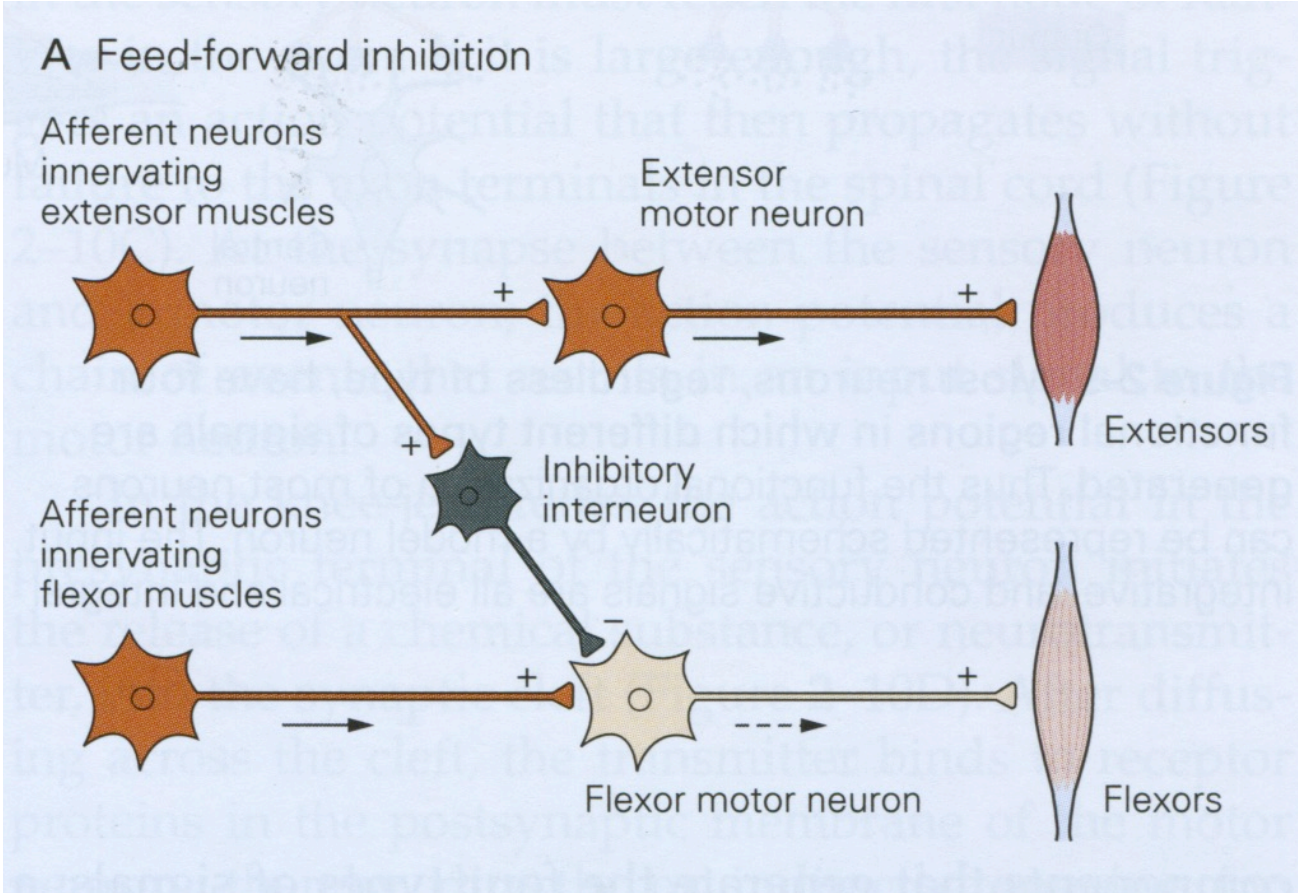
inputs can be positive (+) or negative(-)



while individual neurons are either excitatory or inhibitory, inhibitory interneurons can make an excitatory neuron inhibit indirectly

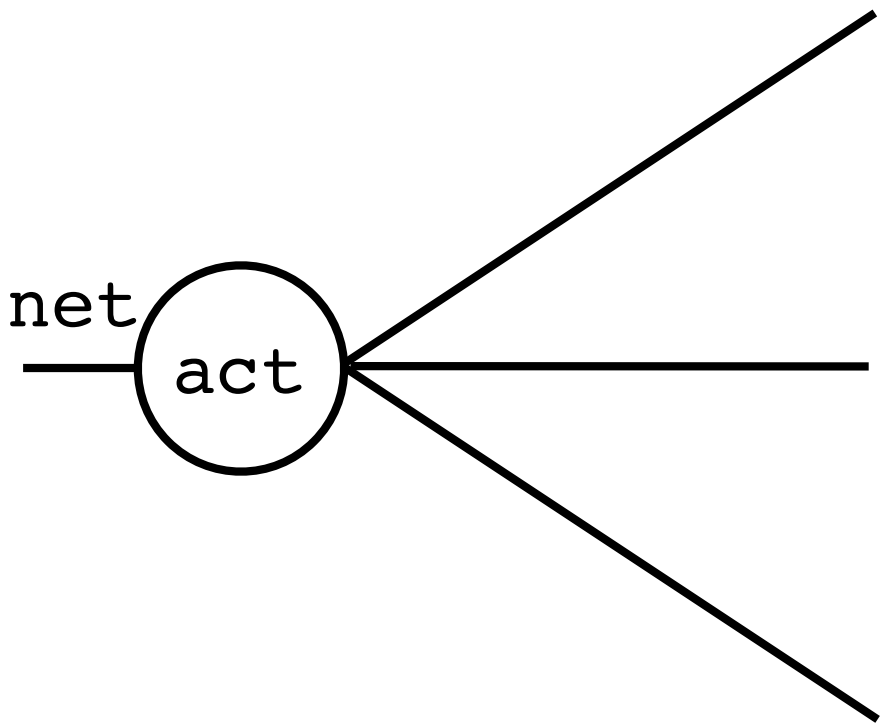
inputs can be positive (+) or negative(-)

so for simplicity, we allow the same unit to be either excitatory or inhibitory

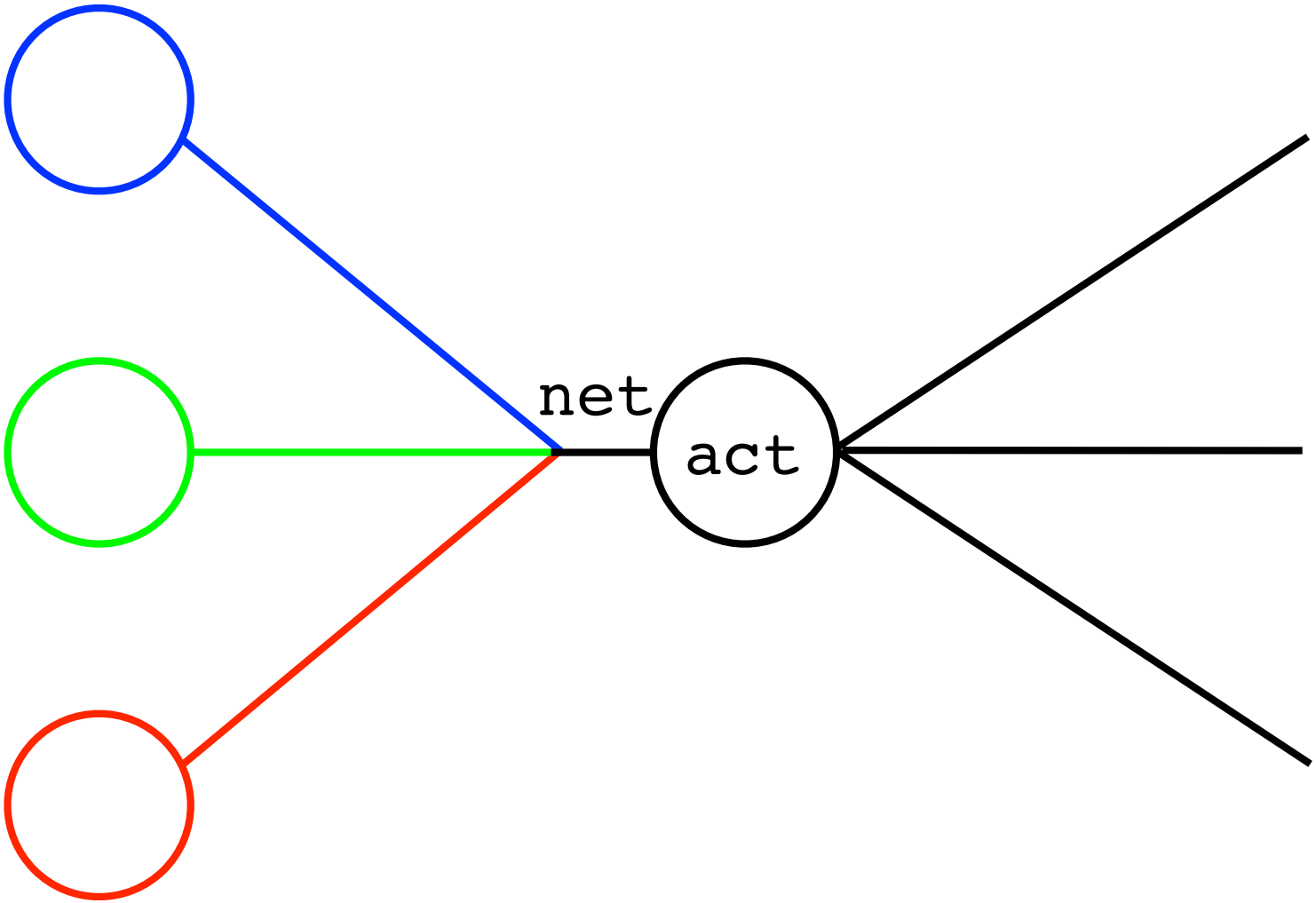


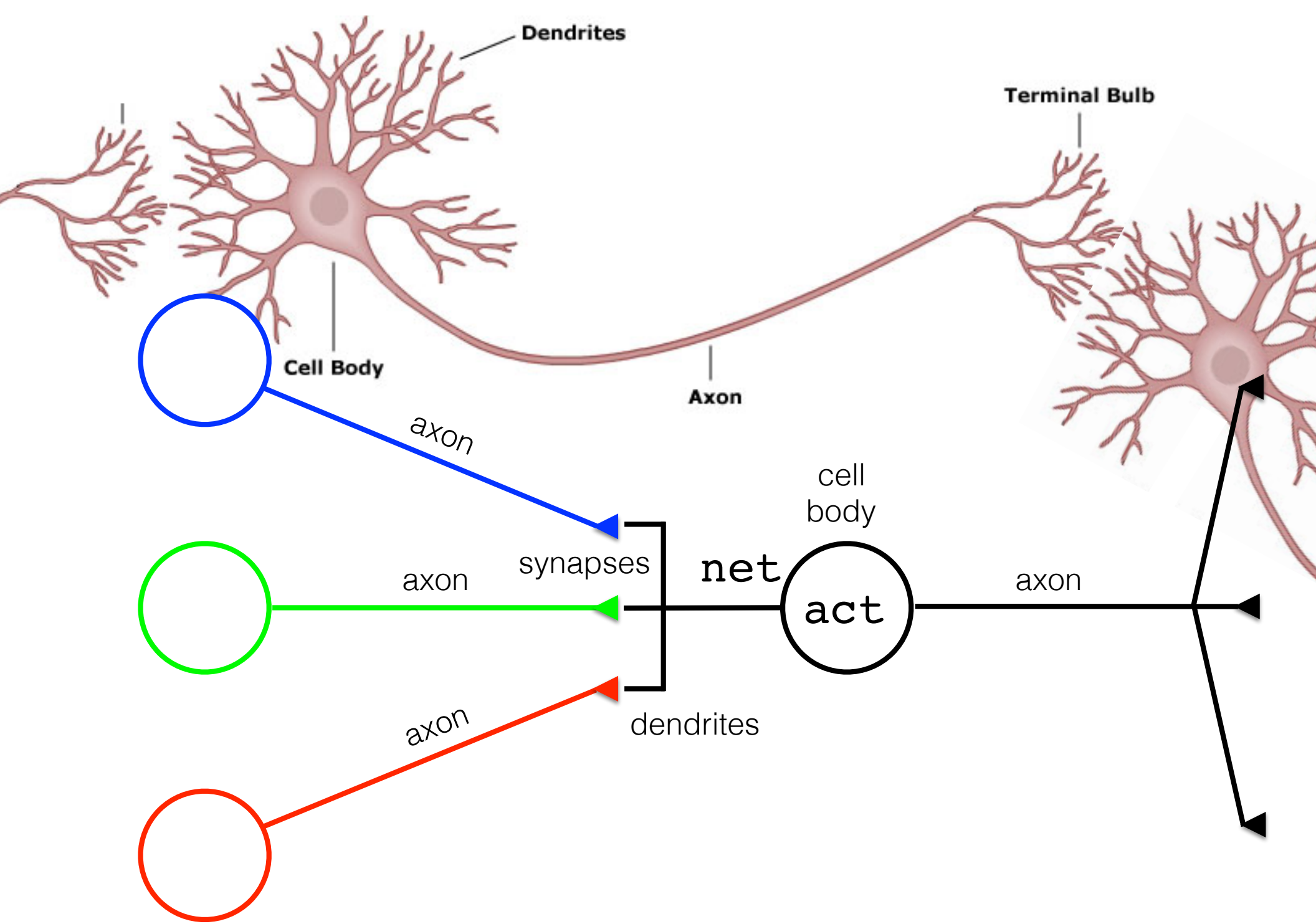
while the implementation is different whether you allow or don't allow the same unit to be excitatory and inhibitory, the kinds of computations a network performs may well be indistinguishable

where does the net input come from?

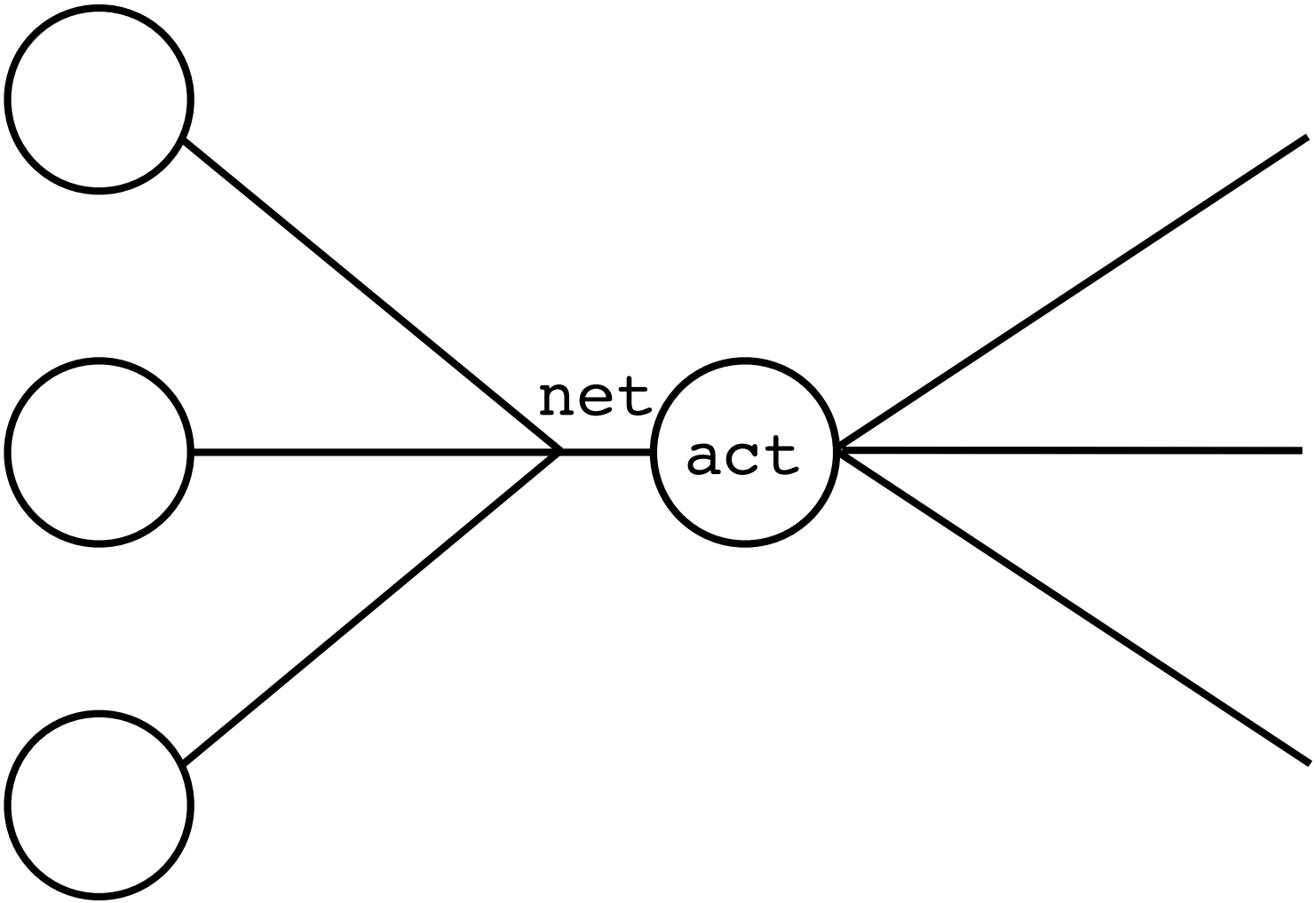


where does the net input come from?



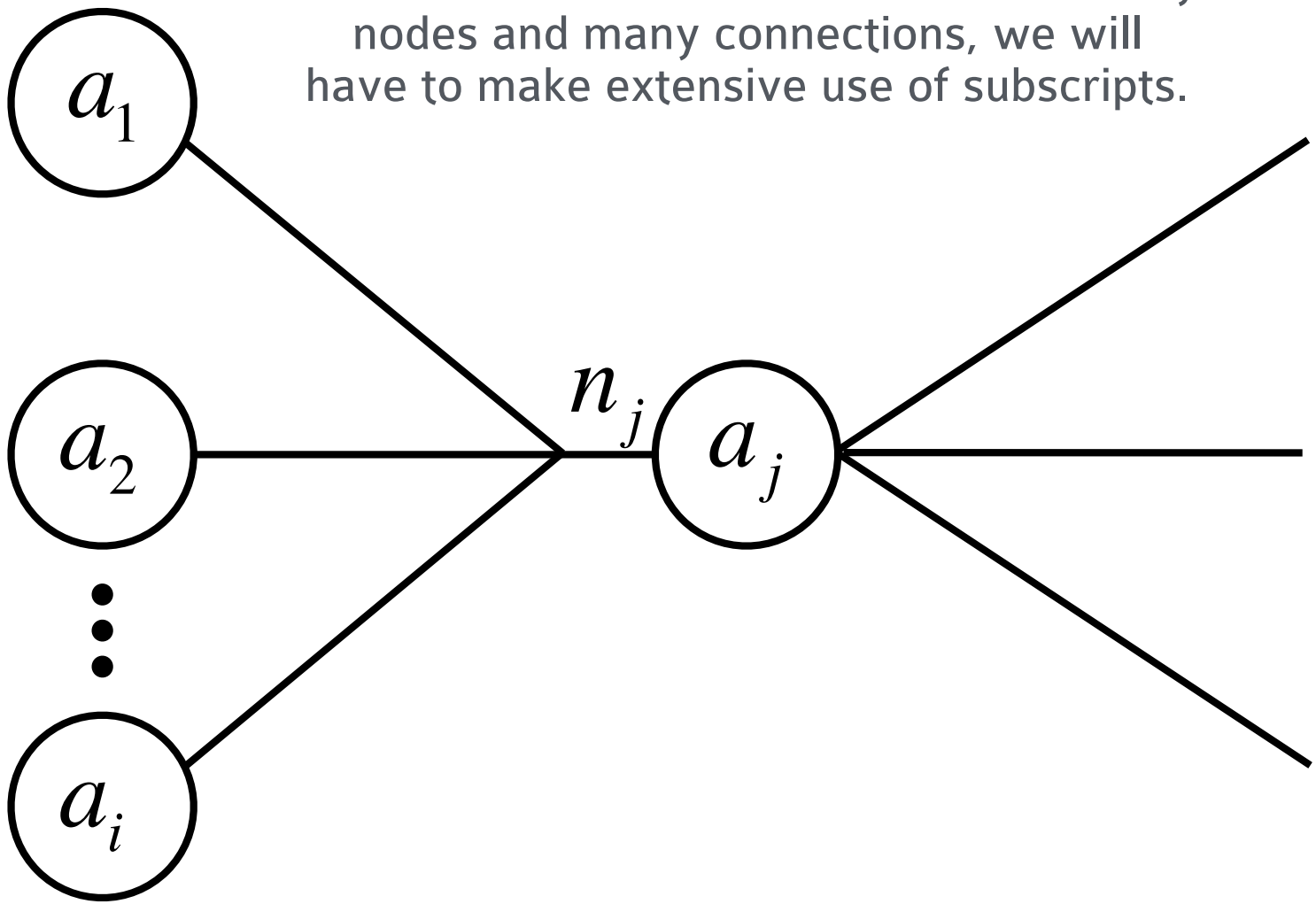

Dendrites
Terminal Bulb
Cell Body
Axon
axon
cell
body
synapses
net
act
axon
axon
axon
dendrites

where does the net input come from?



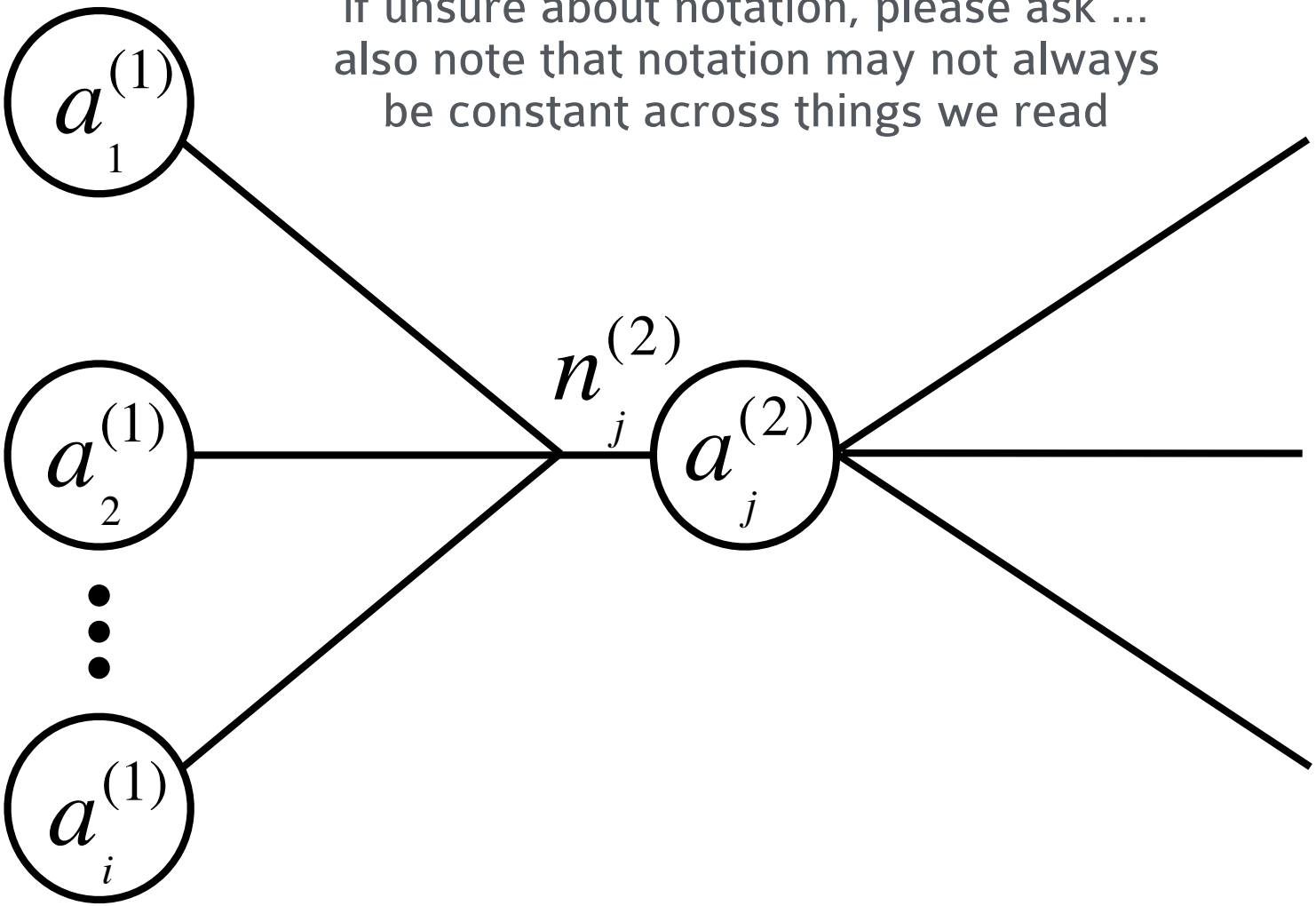
activation of pre-synaptic neurons

Note: because neural networks have many nodes and many connections, we will have to make extensive use of subscripts.

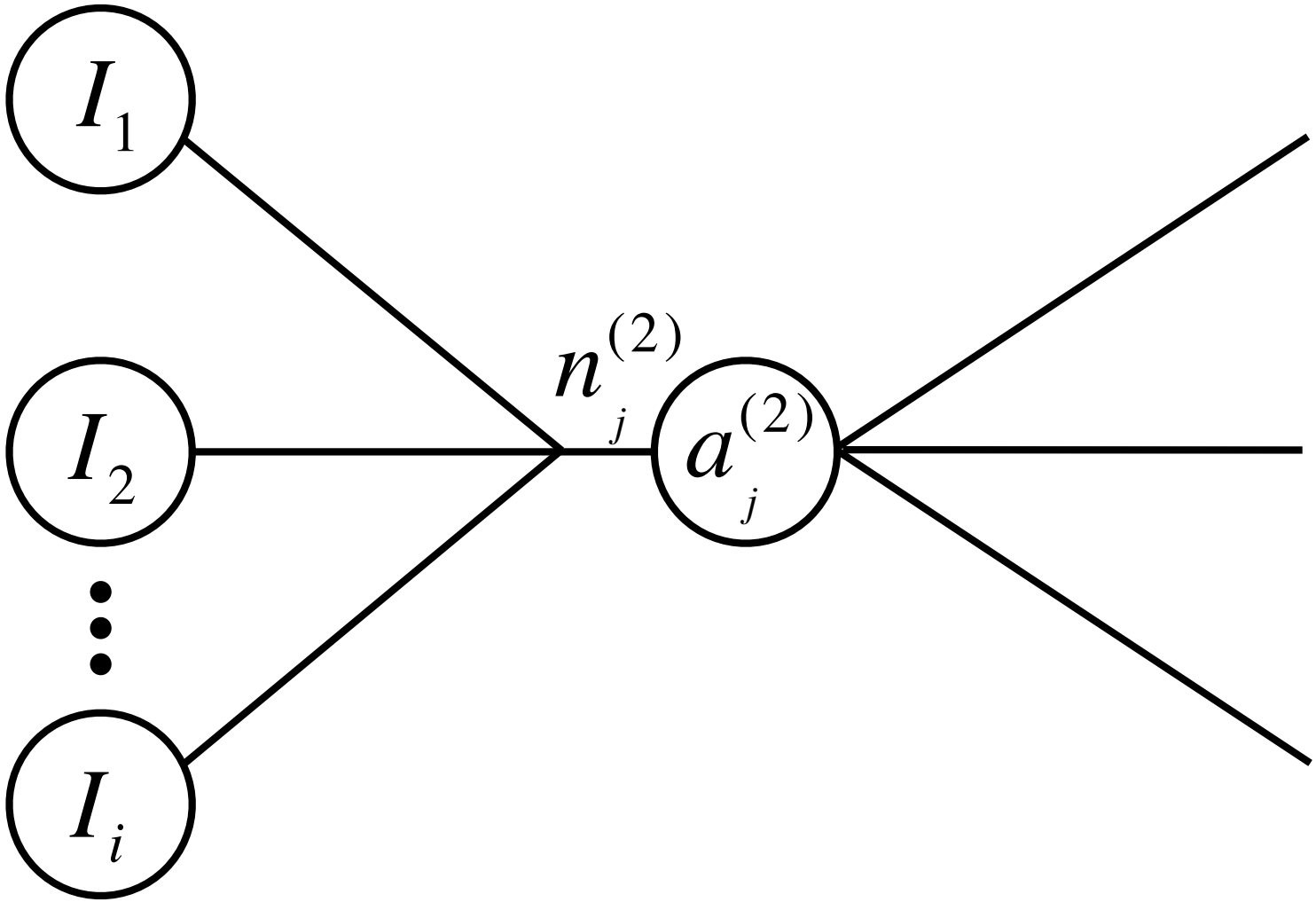


activation of pre-synaptic neurons

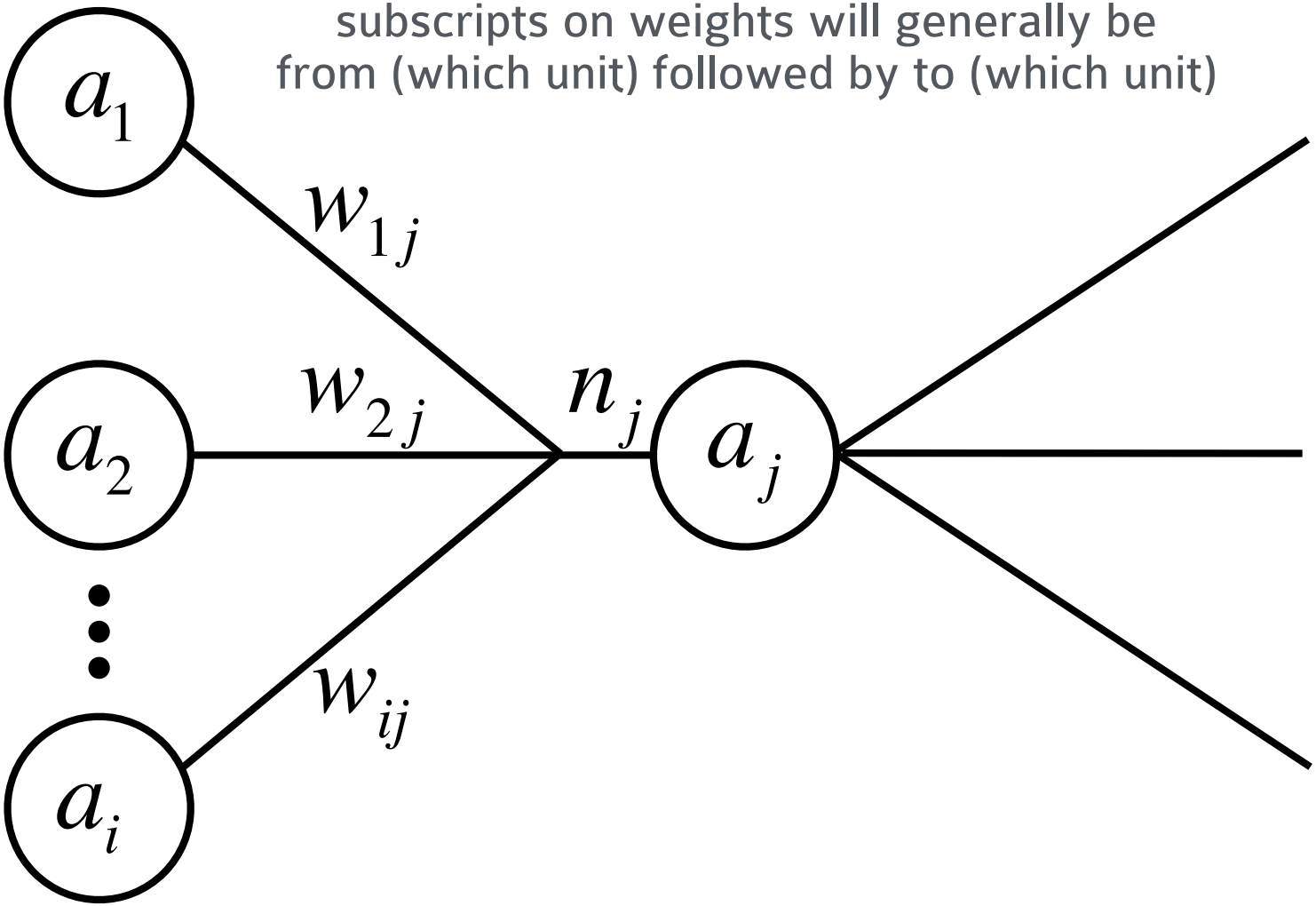
and sometimes superscripts as well ...
if unsure about notation, please ask ...
also note that notation may not always be constant across things we read



inputs clamped
onto network



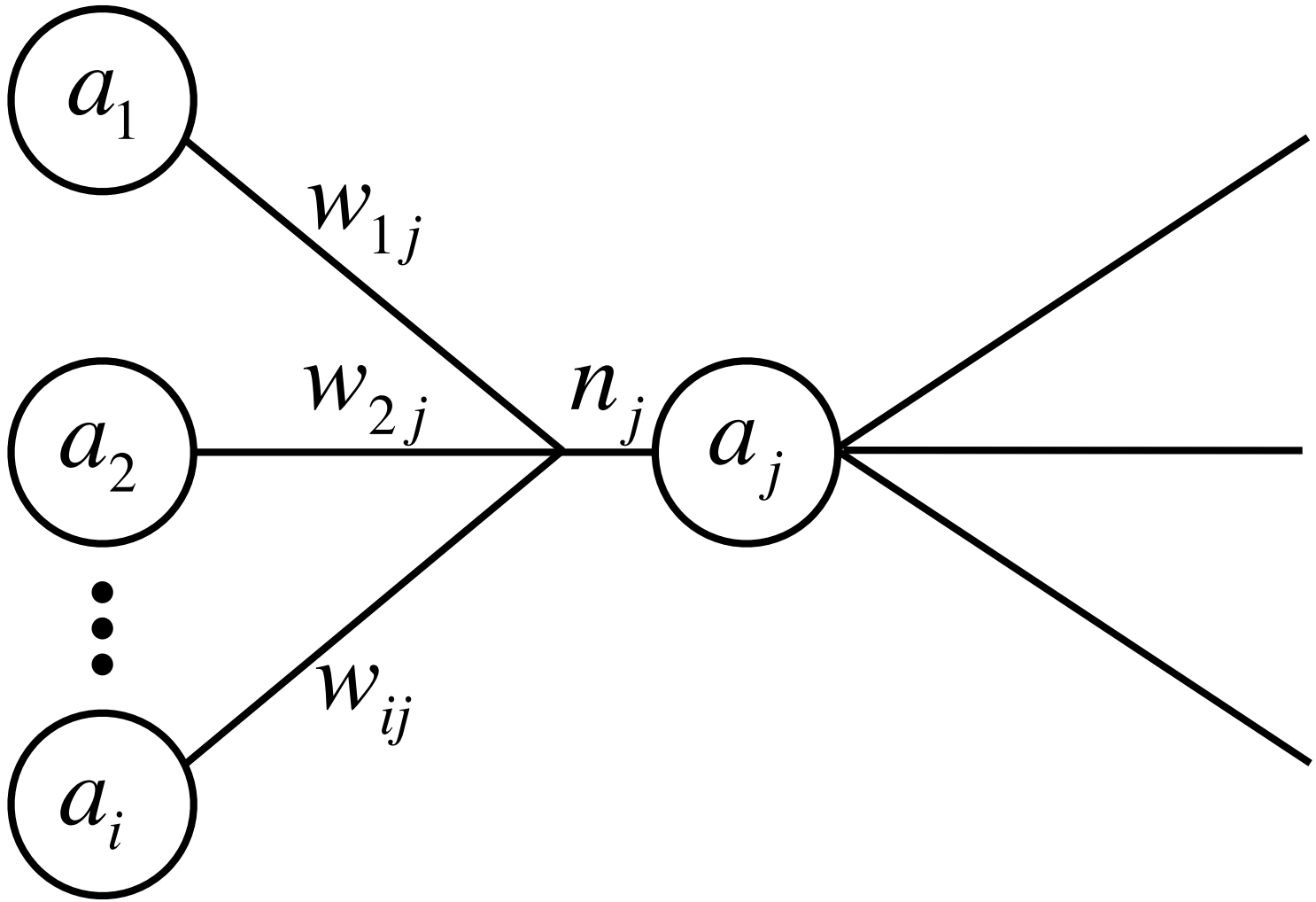
weight of
synaptic
connections



impact of one pre-synaptic neuron

$$a_i w_{ij}$$

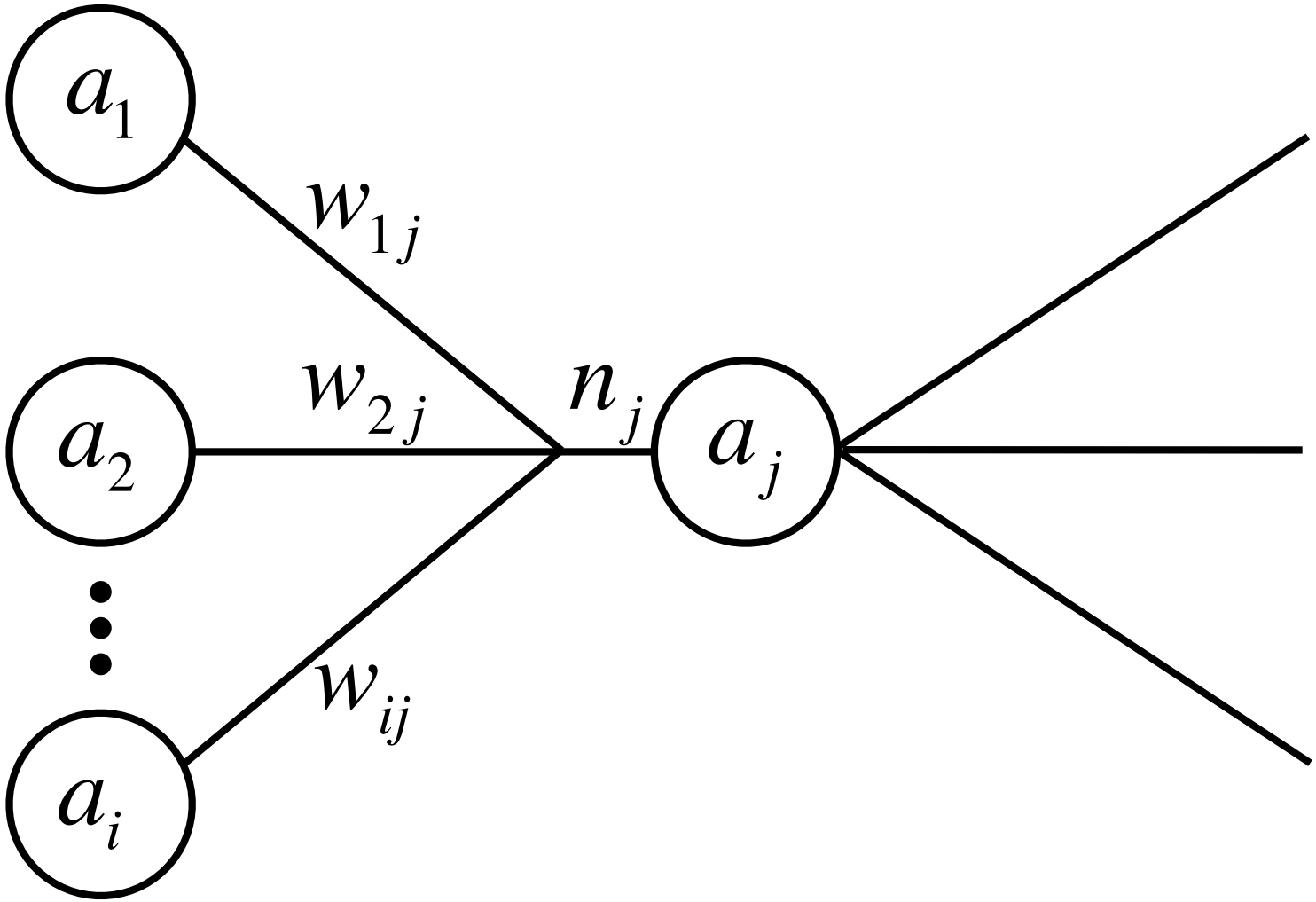
multiplicatively varying strength of connection based on number/strength of synapses



multiplicatively varying strength of connection based on number/strength of synapses

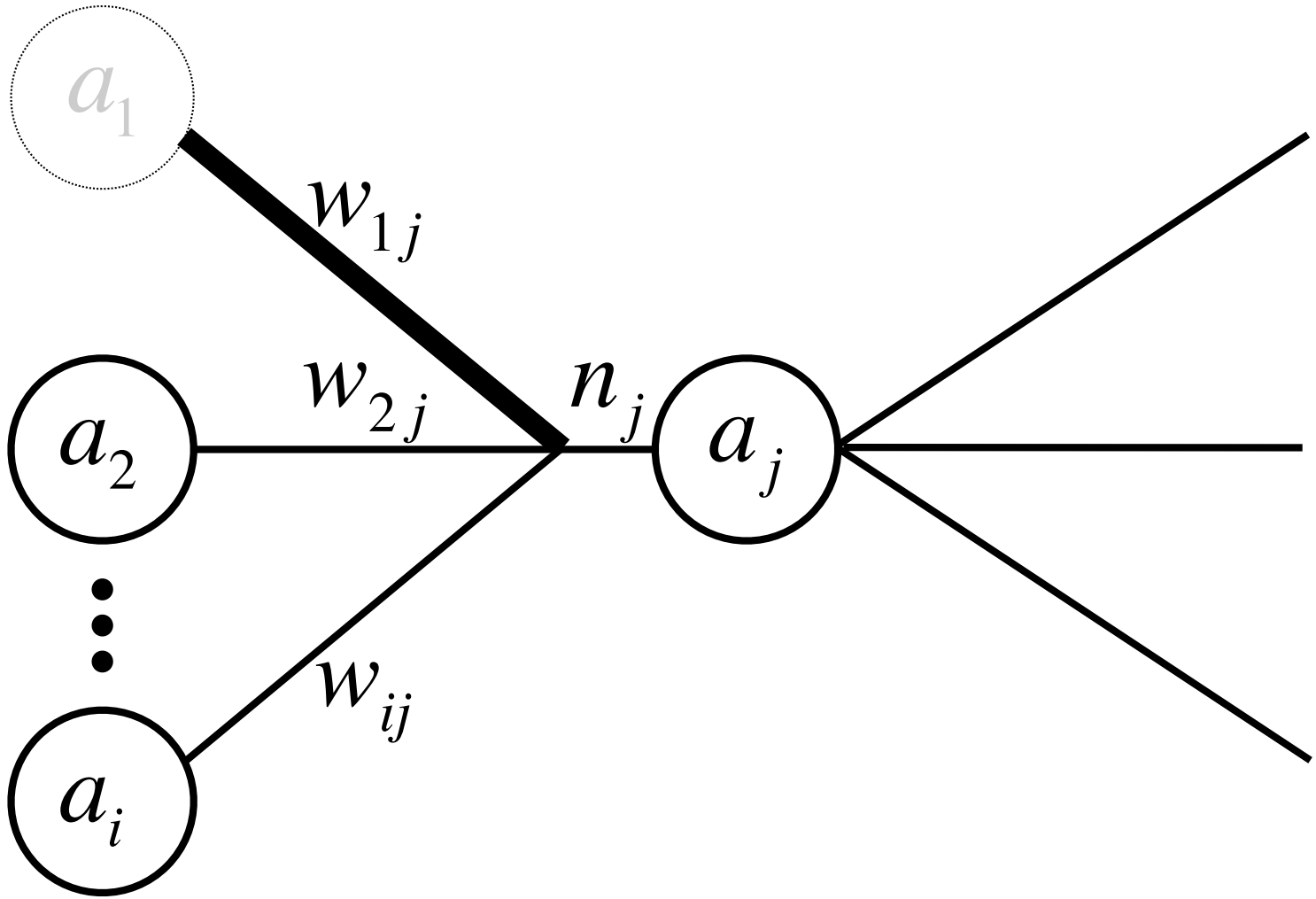
why is it multiplicative and not additive?

multiplication is more then just repeated addition



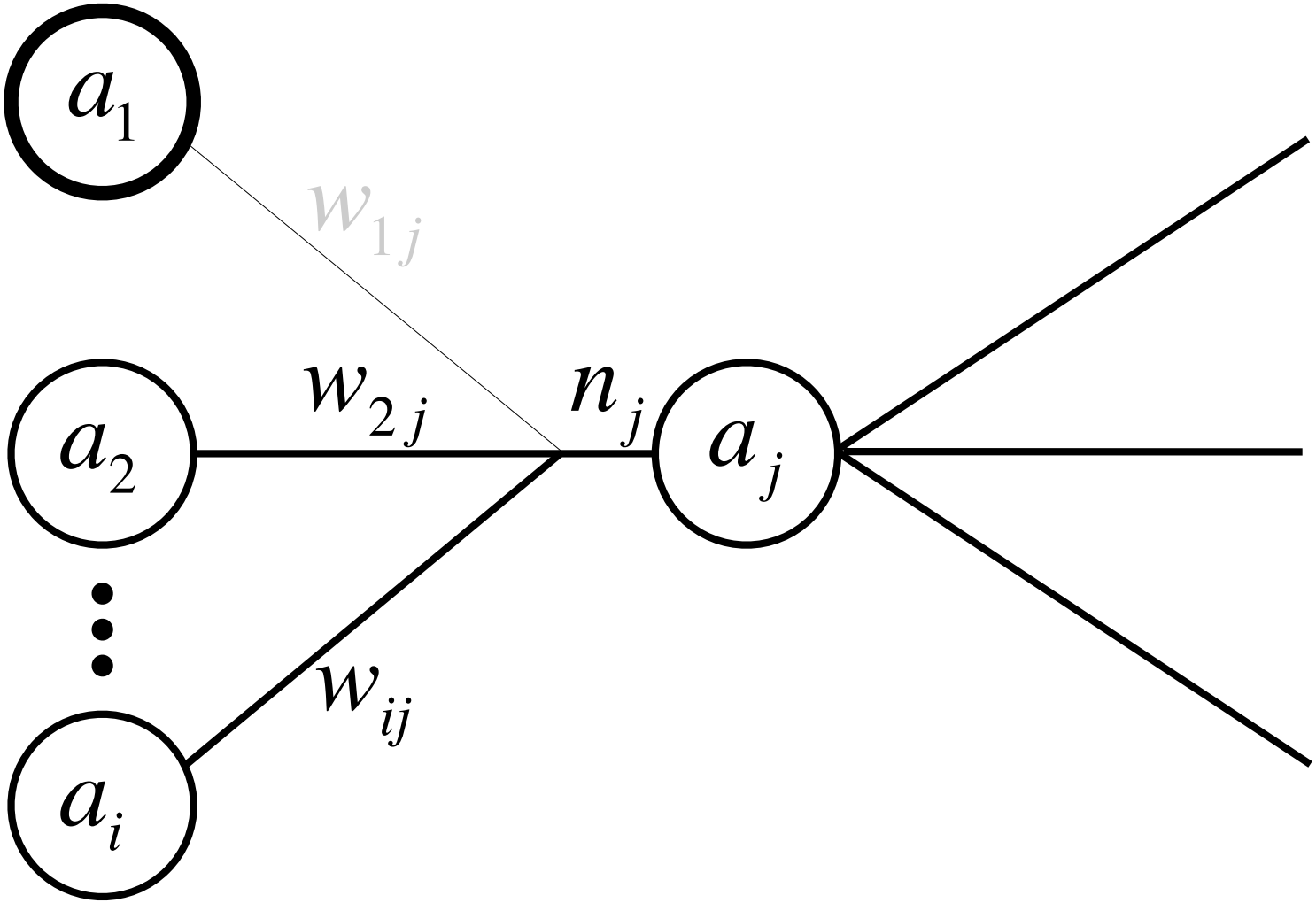
if the pre-synaptic neuron is inactive, it does not matter how strong the connection

multiplicatively varying strength of connection based on number/strength of synapses



if the connection is weak or absent entirely, it does not matter if the pre-synaptic neuron is active or not

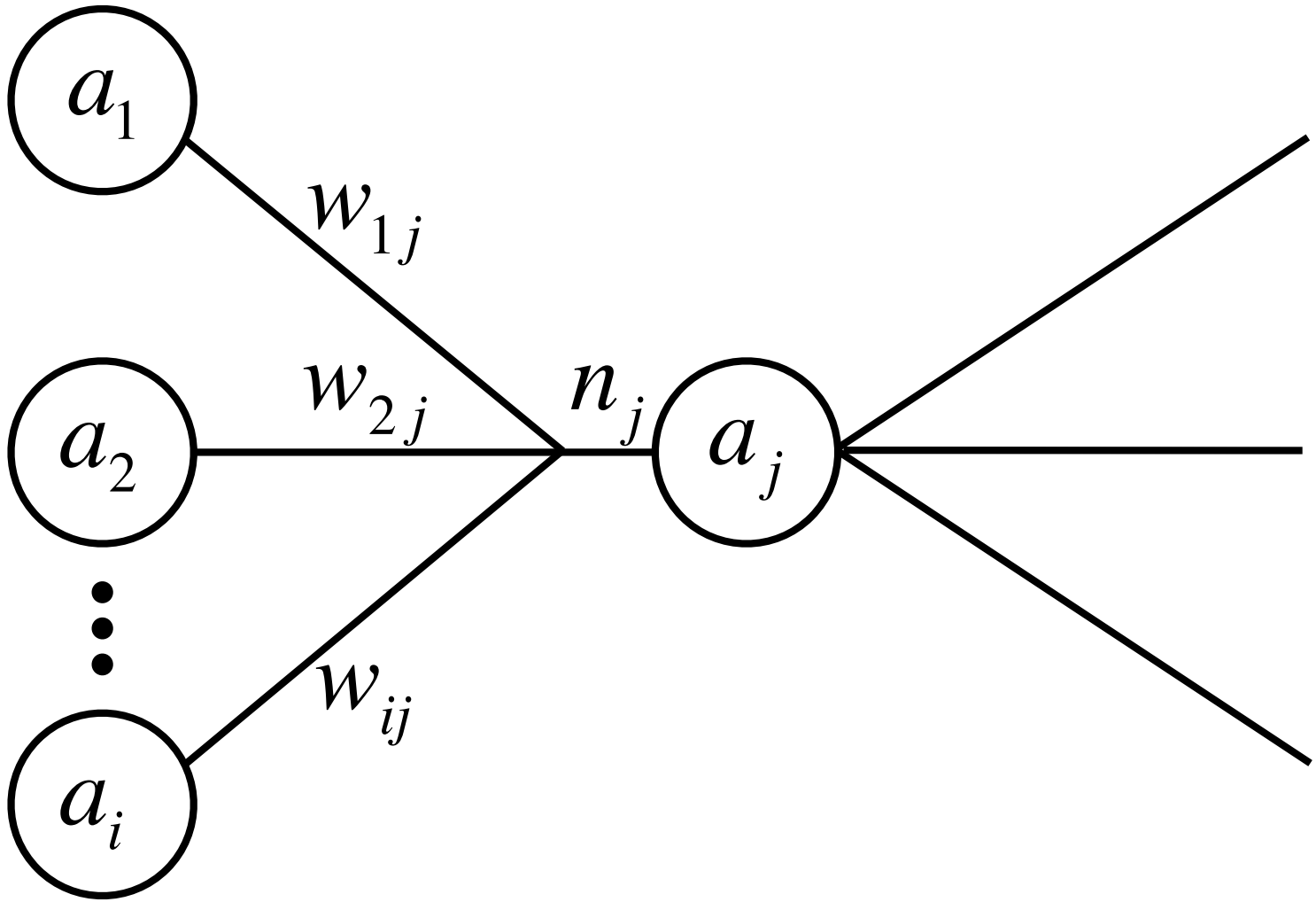
multiplicatively varying strength of connection based on number/strength of synapses



net input sums the weighted inputs

$$n_j = \sum_i a_i w_{ij}$$

inputs integrate at the cell body - they are added together



let's take a look at this equation

$$n_j = \sum_i a_i w_{ij}$$

let's take a look at this equation

$$n_j = \sum_i a_i w_{ij}$$

$$a = \left[a_1, a_2, \ldots, a_m \right]$$ activation of all the input nodes

$$w_j = \left[w_{1j}, w_{2j}, \ldots, w_{mj} \right]$$ all weights going to 2nd layer node j

let's take a look at this equation

$$n_j = \sum_i a_i w_{ij}$$

$$a = \left[a_1, a_2, \ldots, a_m \right]$$

$$w_j = \left[w_{1j}, w_{2j}, \ldots, w_{mj} \right]$$

```
n = 0
for i in np.arange(len(a)):
    n += a[i]*wj[i]
```

let's take a look at this equation

$$n_j = \sum_i a_i w_{ij}$$

$$a = \left[a_1, a_2, \ldots, a_m \right]$$

$$w_j = \left[w_{1j}, w_{2j}, \ldots, w_{mj} \right]$$

```
n = sum(a*wj)
```

element-wise multiplication of numpy arrays

(different from Matlab, which requires .* operator)

let's take a look at this equation

$$n_j = \sum_i a_i w_{ij}$$

$$a = \left[a_1, a_2, \ldots, a_m \right]$$

$$w_j = \left[w_{1j}, w_{2j}, \ldots, w_{mj} \right]$$

$$n_j = a \cdot w_j$$

```
import numpy as np
n = np.dot(a,wj)
```

dot product