Neuro4ML

Neuroscience for machine learners

Developed by

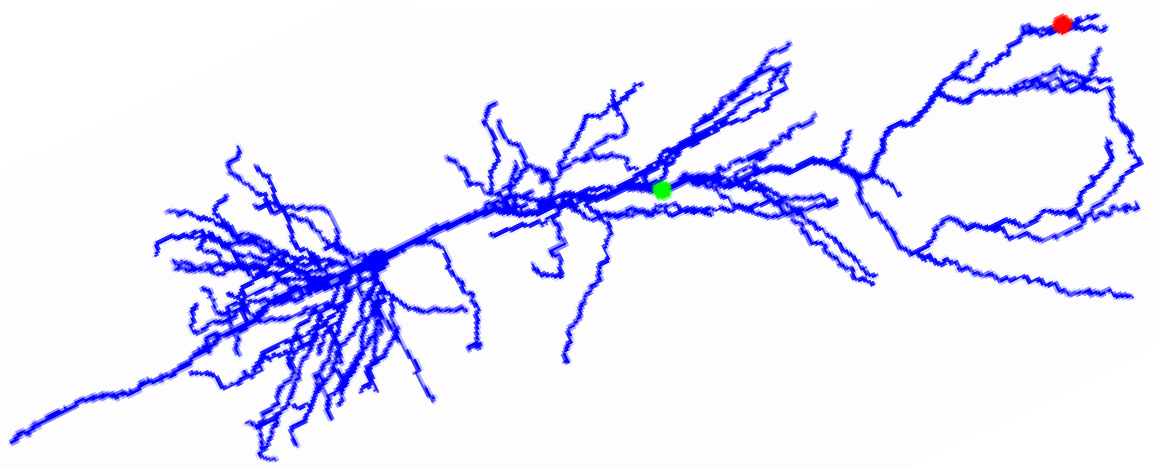
Dan Goodman and Marcus Ghosh

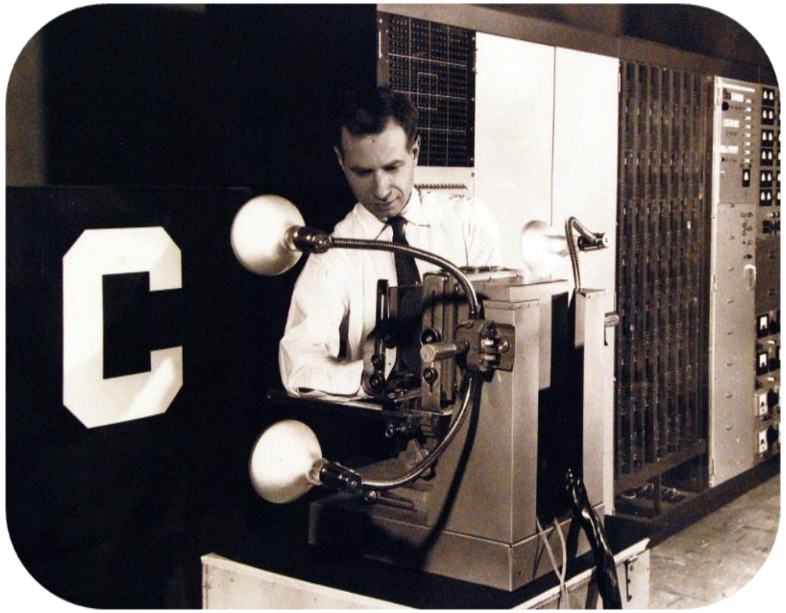
Imperial College London

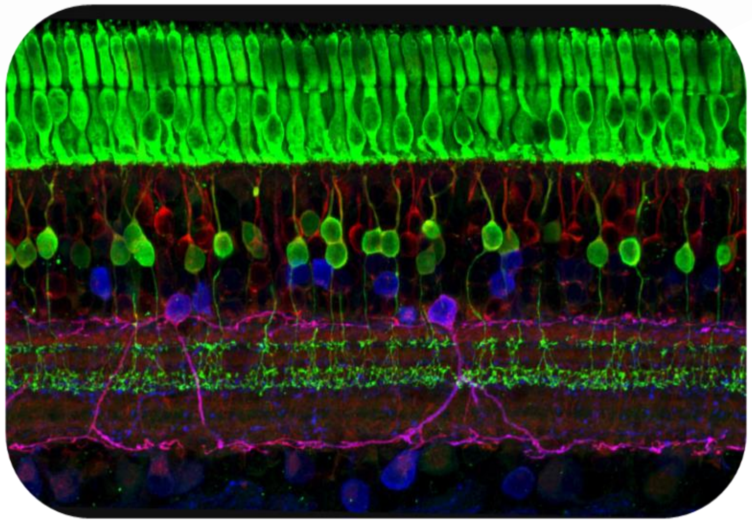
https://neural-reckoning.org

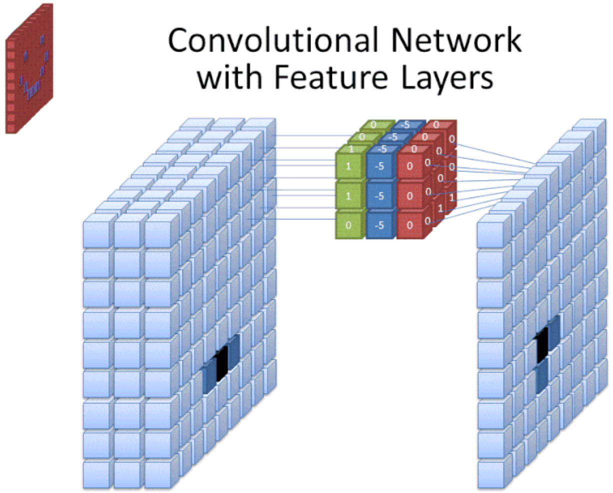
Why neuroscience?

Neuroscience ❤ Machine Learning

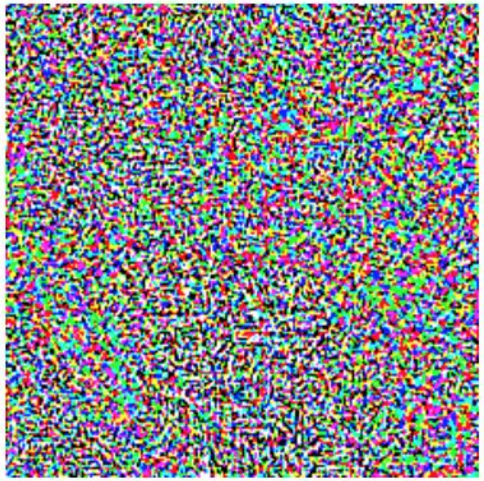








$$+\ .007\ \times$$



$$=$$

$\boldsymbol{x}$

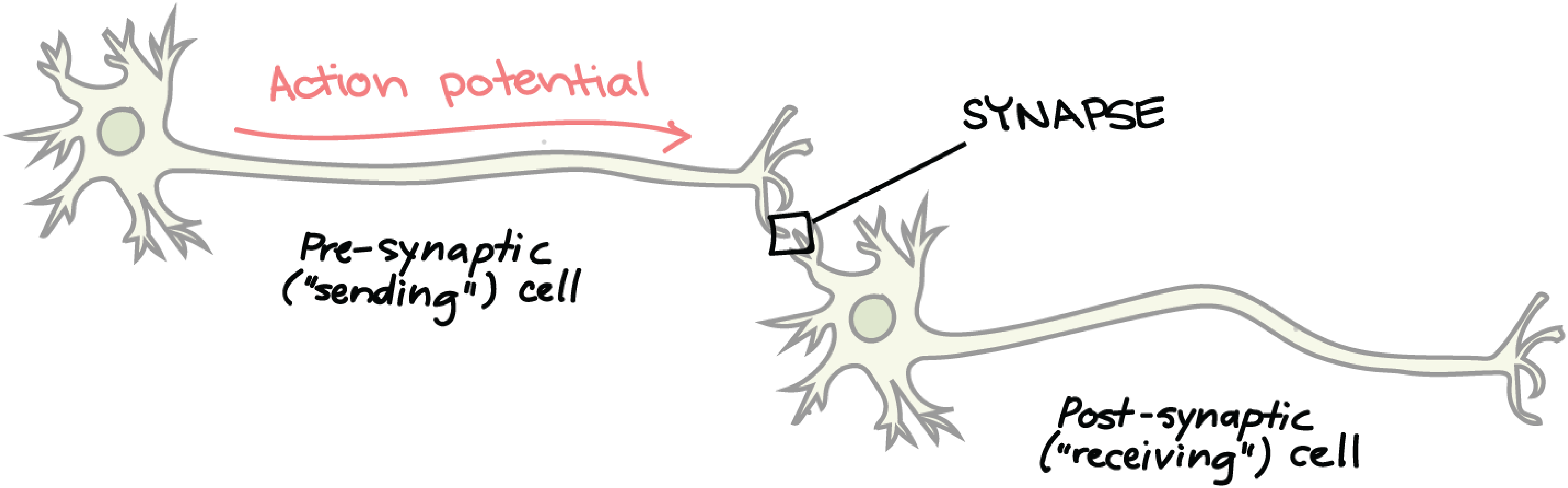
"panda"
57.7% confidence

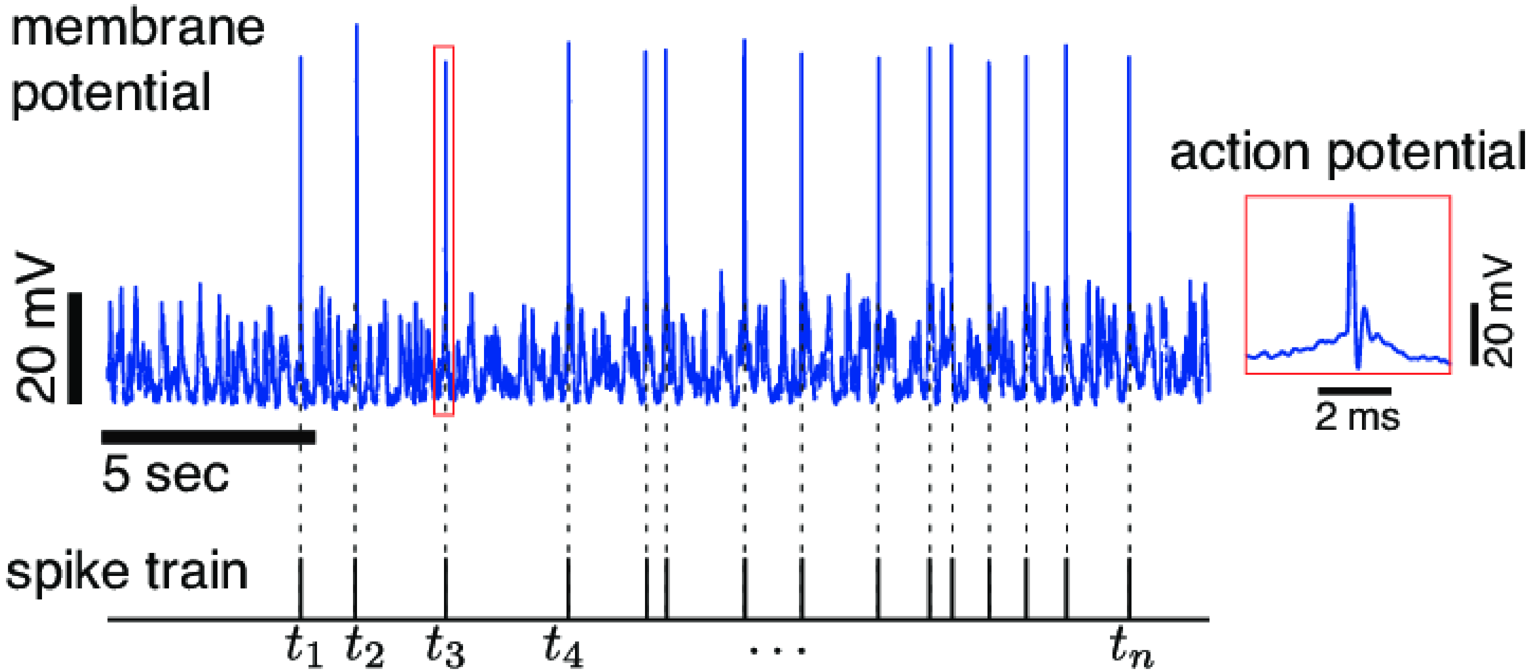
$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$\boldsymbol{x} +$
$\epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence


Action potential
SYNAPSE
Pre-synaptic
("sending") cell
Post-synaptic
("receiving") cell


membrane
potential
20 mV
5 sec
action potential
20 mV
2 ms
spike train
t_1 t_2 t_3 t_4 . . . t_n

3-surrogate-sound-localisation-solution.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text Copy to Drive

Sound localisation with surrogate gradient descent

In this notebook, we're going to use surrogate gradient descent to find a solution to the sound localisation problem we solved by hand in the previous notebook. The surrogate gradient descent approach and code is heavily inspired by (certainly not stolen) from [Friedemann Zenke's SPyTorch tutorial](), which I recommend for a deeper dive into the maths.

```
[ ] import os

    import numpy as np
    import matplotlib.pyplot as plt
    from matplotlib.gridspec import GridSpec

    import torch
    import torch.nn as nn

    dtype = torch.float

    # Check whether a GPU is available
    if torch.cuda.is_available():
        device = torch.device("cuda")
    else:
        device = torch.device("cpu")

    my_computer_is_slow = True # set this to True if using Colab
```

Sound localisation stimuli

The following function creates a set of stimuli that can be used for training or testing. We have two e
the signal delayed by an IPD we can write as α in equations (`ipd` in code). The basic signal is a sine
positive, so $(1/2)(1+\sin(\theta))$. In addition, for each ear there will be N_a neurons per ear (`anf_per_`
fibres). Each neuron generates Poisson spikes at a certain firing rate, and these Poisson spike trains
hard to train delays, we seed it with uniformly distributed delays from a minimum of 0 to a maximum of $\pi/2$ in each ear, so that the differences
between the two ears can cover the range of possible IPDs ($-\pi/2$ to $\pi/2$). We do this directly by adding a phase delay to each neuron. So for

Interested in doing a PhD? See
http://neural-reckoning.org/apply_phd.html