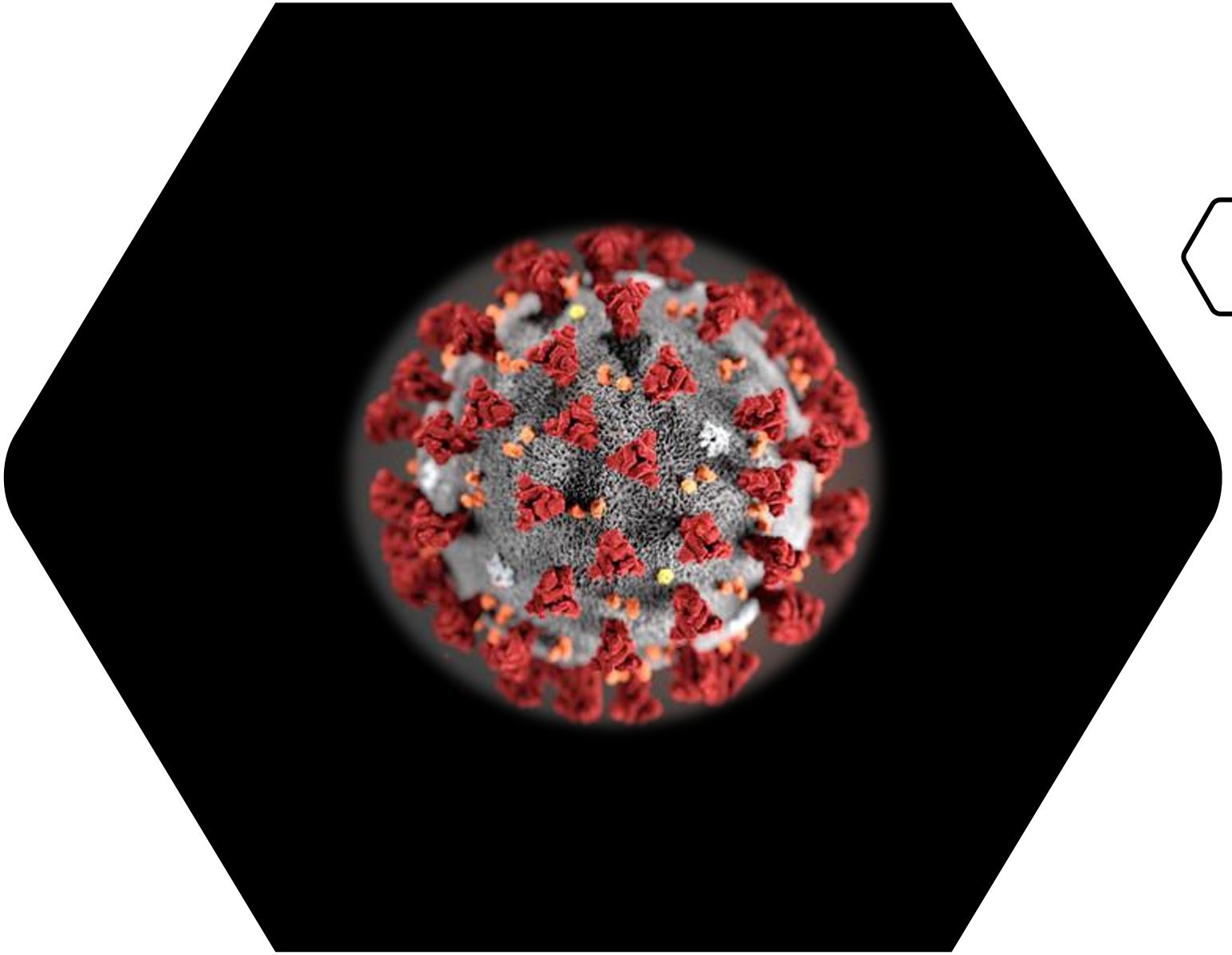
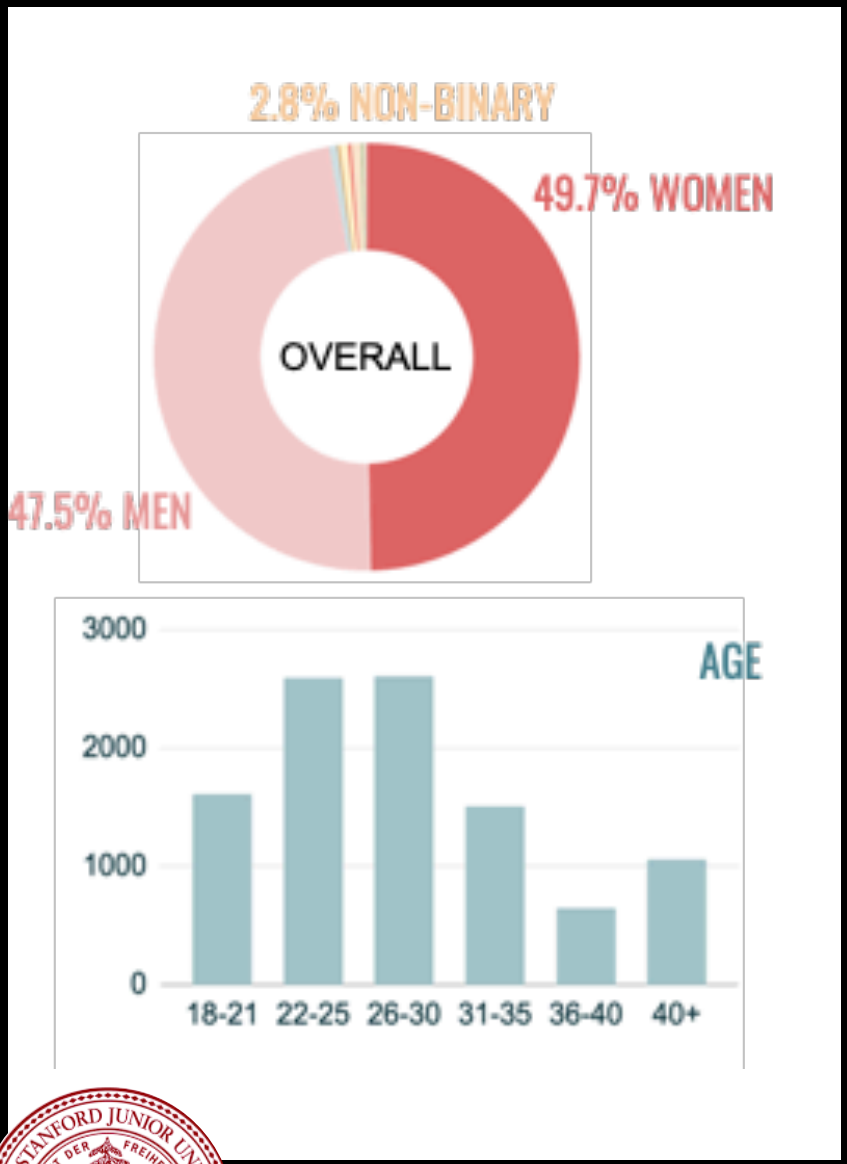
Beyond CS106A

Chris Piech
CS106A, Stanford University

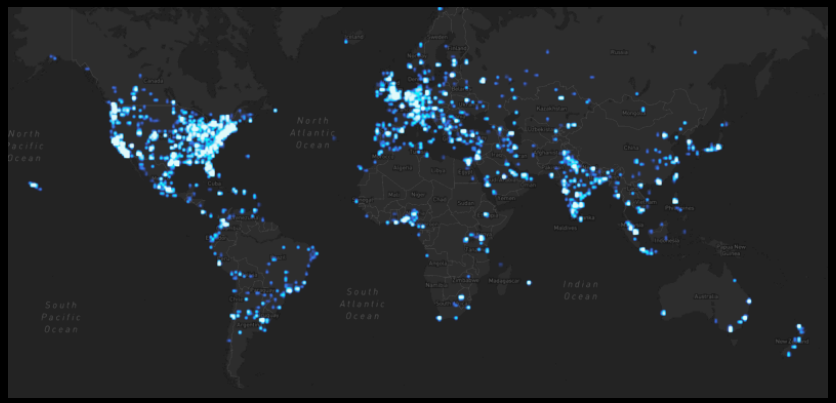
Did you know?



Code in Place : Course with the most section leaders?



905 section leaders teach
10,000 students
First half of Stanford CS106A



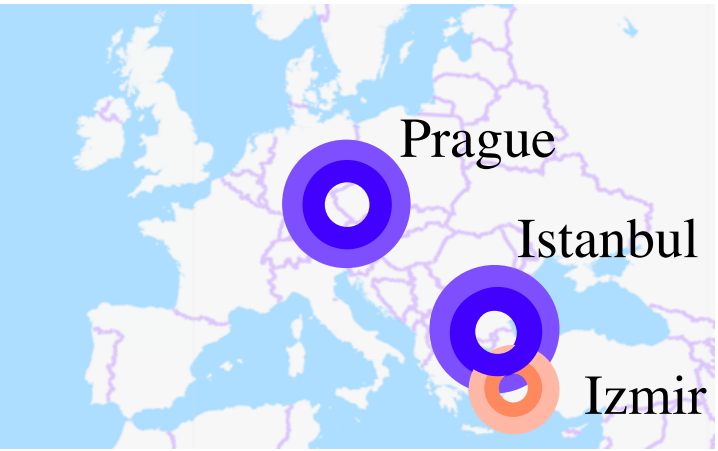
20% experienced job loss or home loss
10x retention vs baseline MOOC
99% wanted more (after first section)
6k hours of live teaching
60k hours of lecture watched
Better CS106A for **Stanford** students

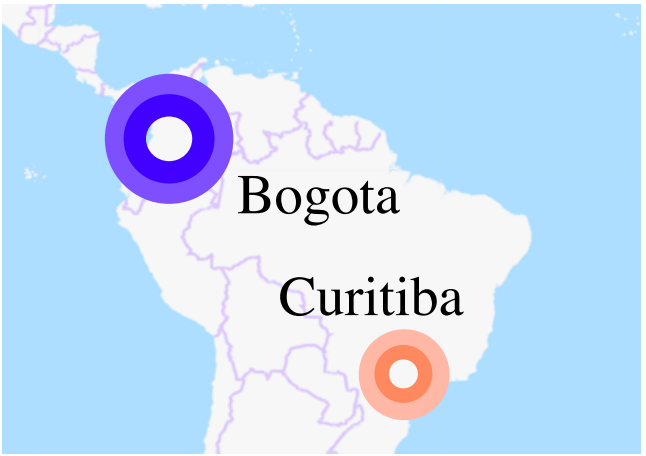
Have you thought about teaching?

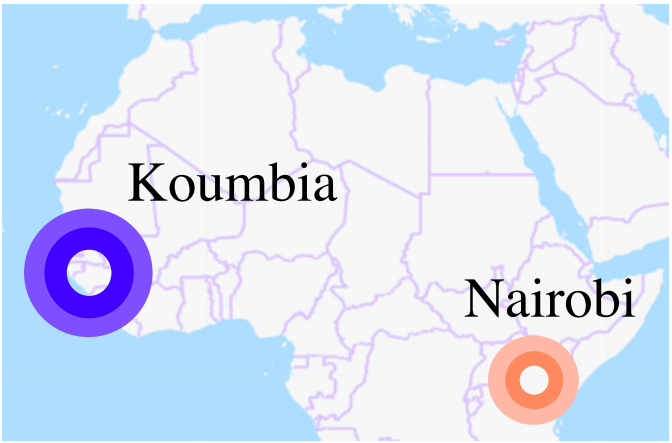
Great way to learn

Great way to give back

In-person teaching around the world.



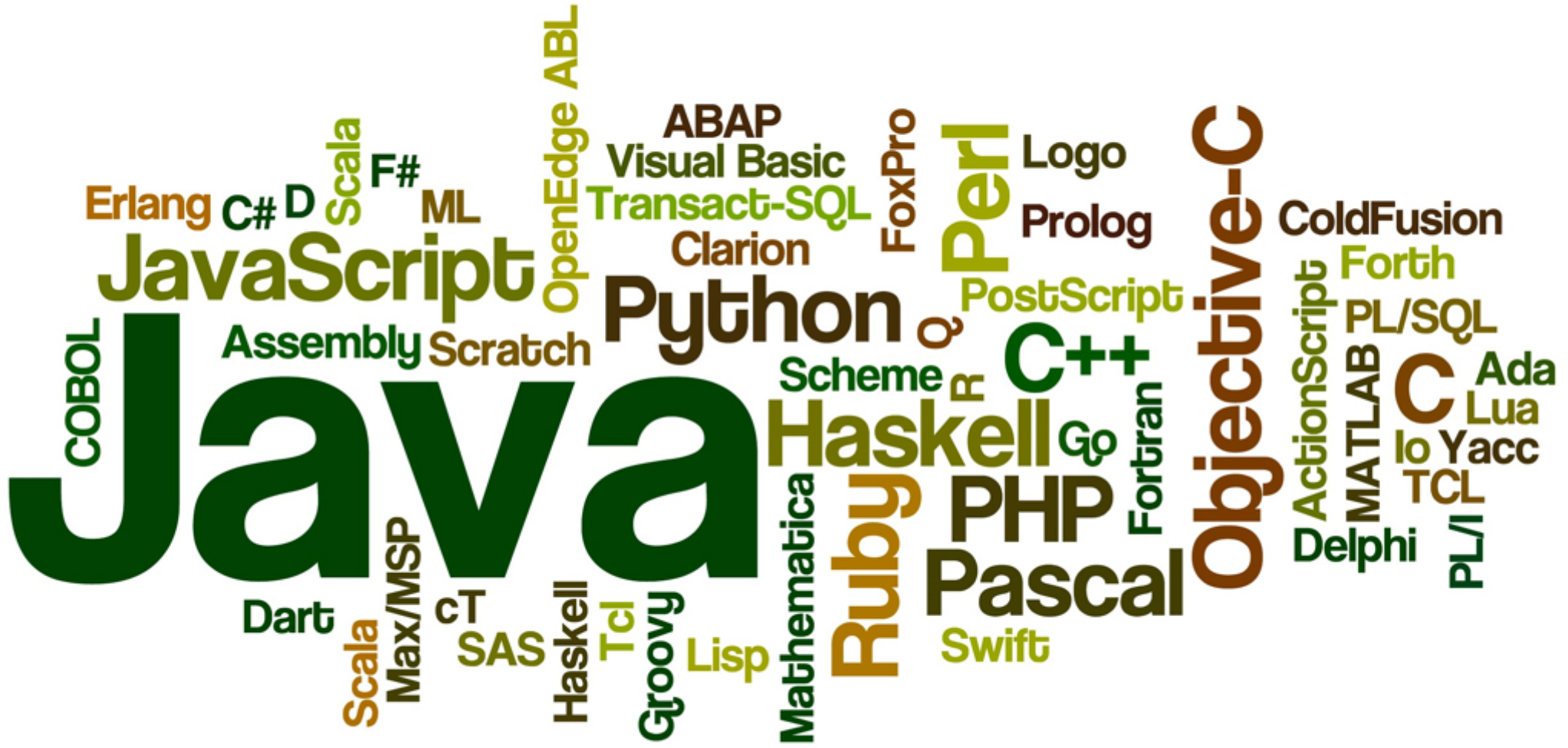




1. How to make your own project
2. What other languages look like
3. Deep Learning in Python

Life after CS106A

Programming Languages



Python

```
evens = []
for i in range(100):
    if i % 2 == 0:
        evens.append(i)
print evens
```

prints [2, 4, 6, 8, 10, 12, …]

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

prints [2, 4, 6, 8, 10, 12, …]

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

prints [2, 4, 6, 8, 10, 12, …]

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

```
prints [2, 4, 6, 8, 10, 12, … ]
```

C++

```
Vector<double> evens;
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
cout << evens << endl;
```

prints [2, 4, 6, 8, 10, 12, …]

Java

```
ArrayList<Double> evens = new ArrayList<Double>();
for(int i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.add(i);
    }
}
println(evens);
```

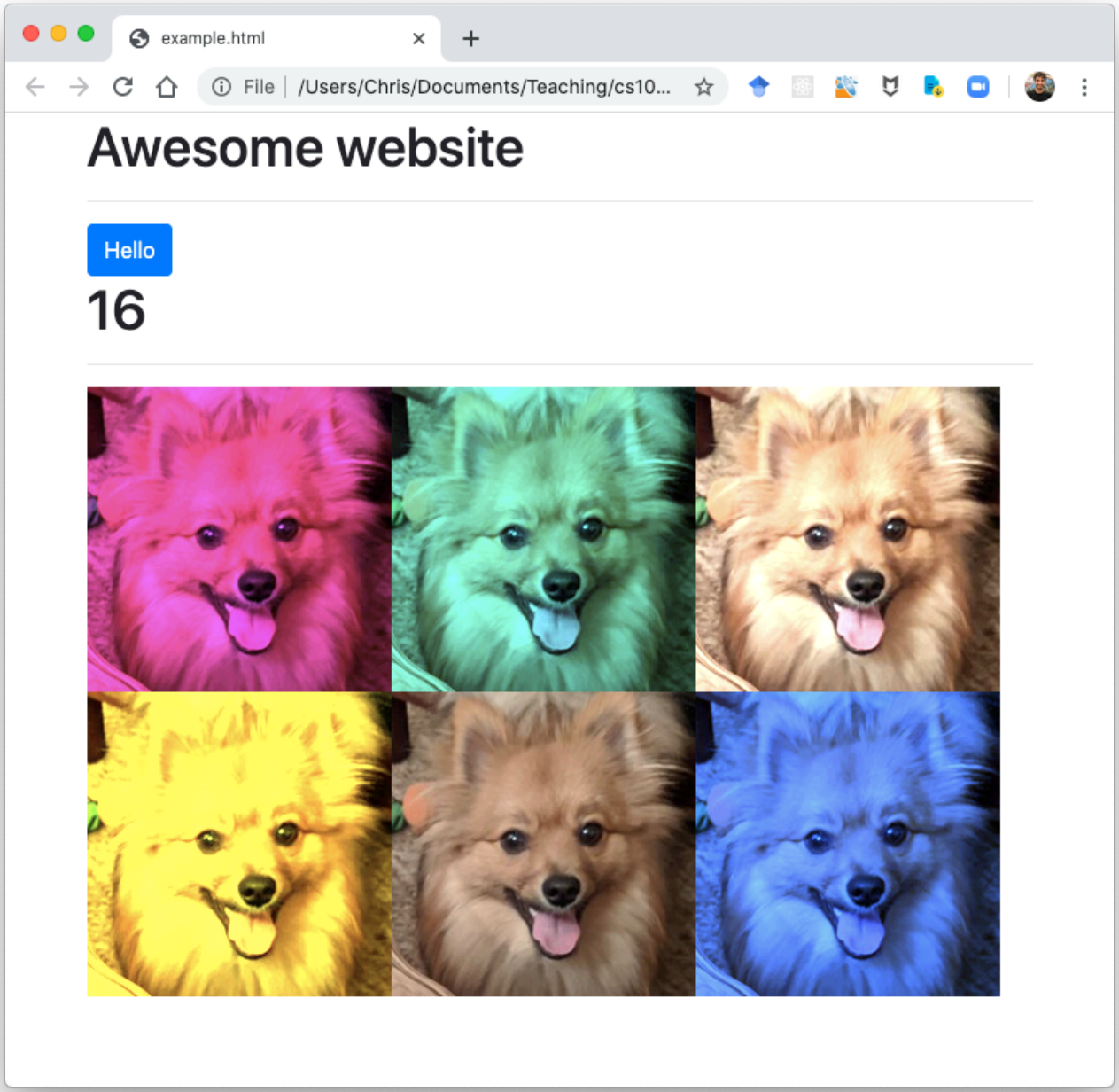
prints [2, 4, 6, 8, 10, 12, …]

Javascript

```
var evens = []
for(var i = 0; i < 100; i++) {
    if(i % 2 == 0) {
        evens.push(i)
    }
}
console.log(evens)
```

prints [2, 4, 6, 8, 10, 12, …]

Lets Play


example.html
File | /Users/Chris/Documents/Teaching/cs10...
Awesome website
Hello
16

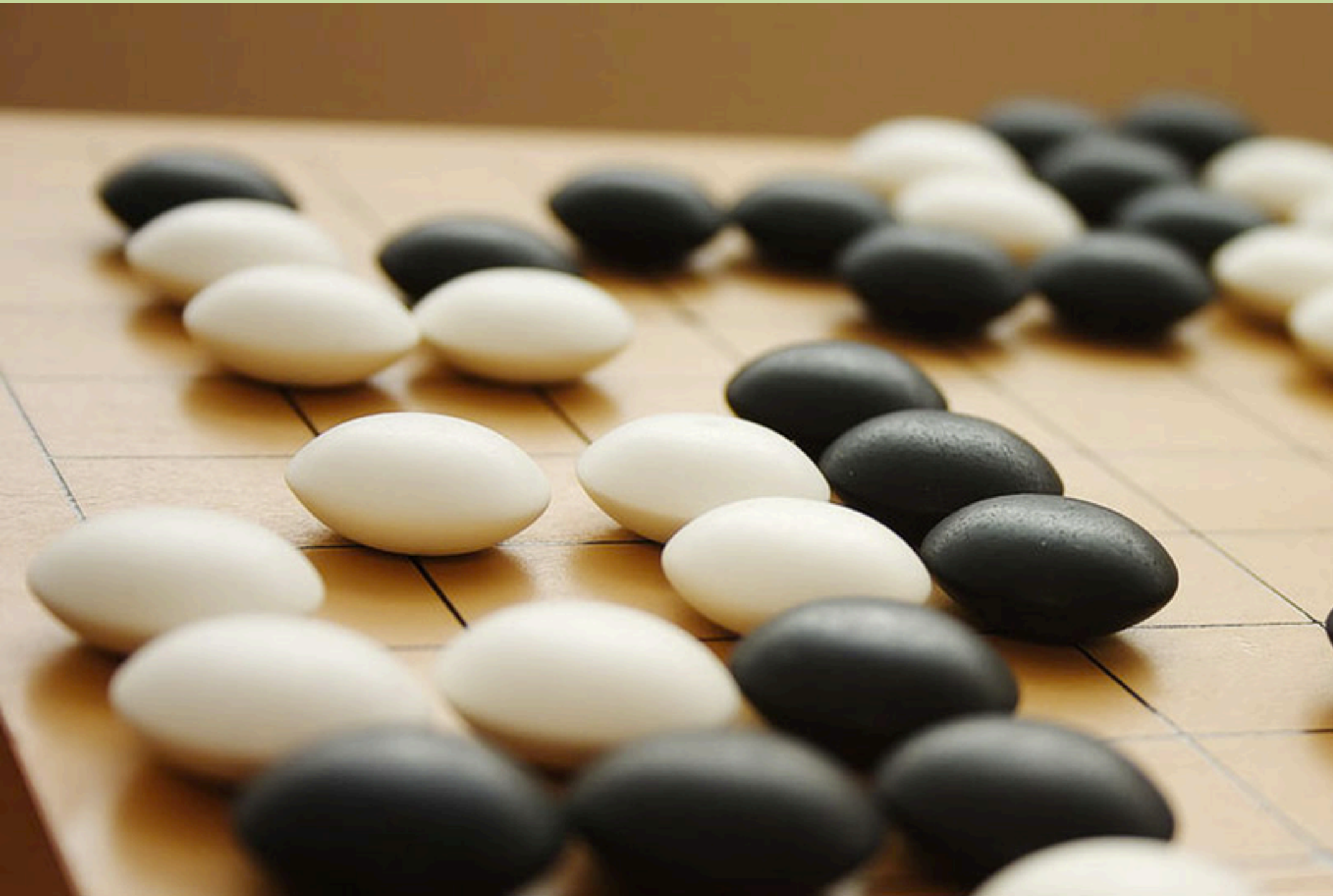
There is something going on
in the world of AI

[suspense]

Self Driving Cars

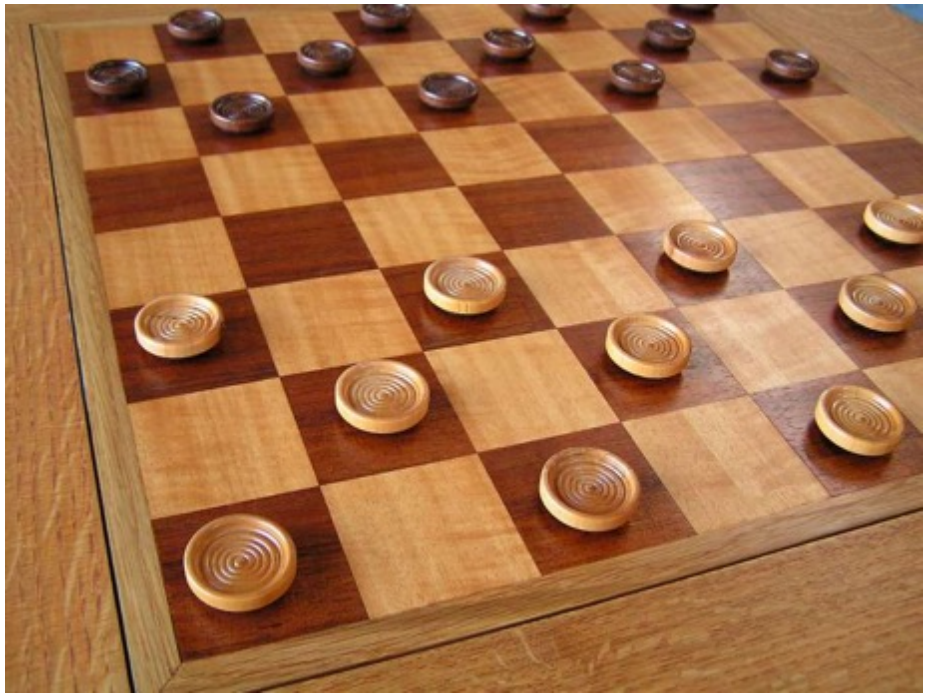
Computers Making Art

The Last Remaining Board Game

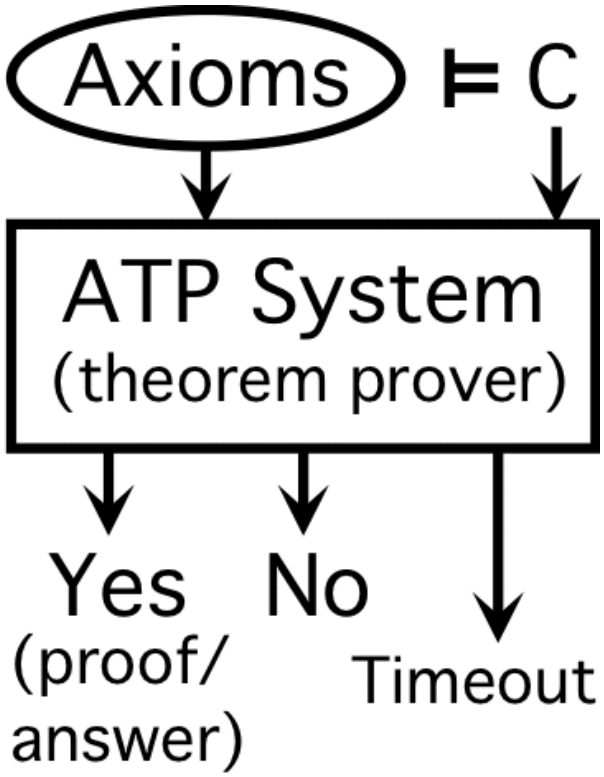


Early Optimism 1950

1952



1955



Computer Vision

Classification

That is a picture of a **one**

Classification

That is a picture of a **zero**

Classification

That is a picture of an **zero**

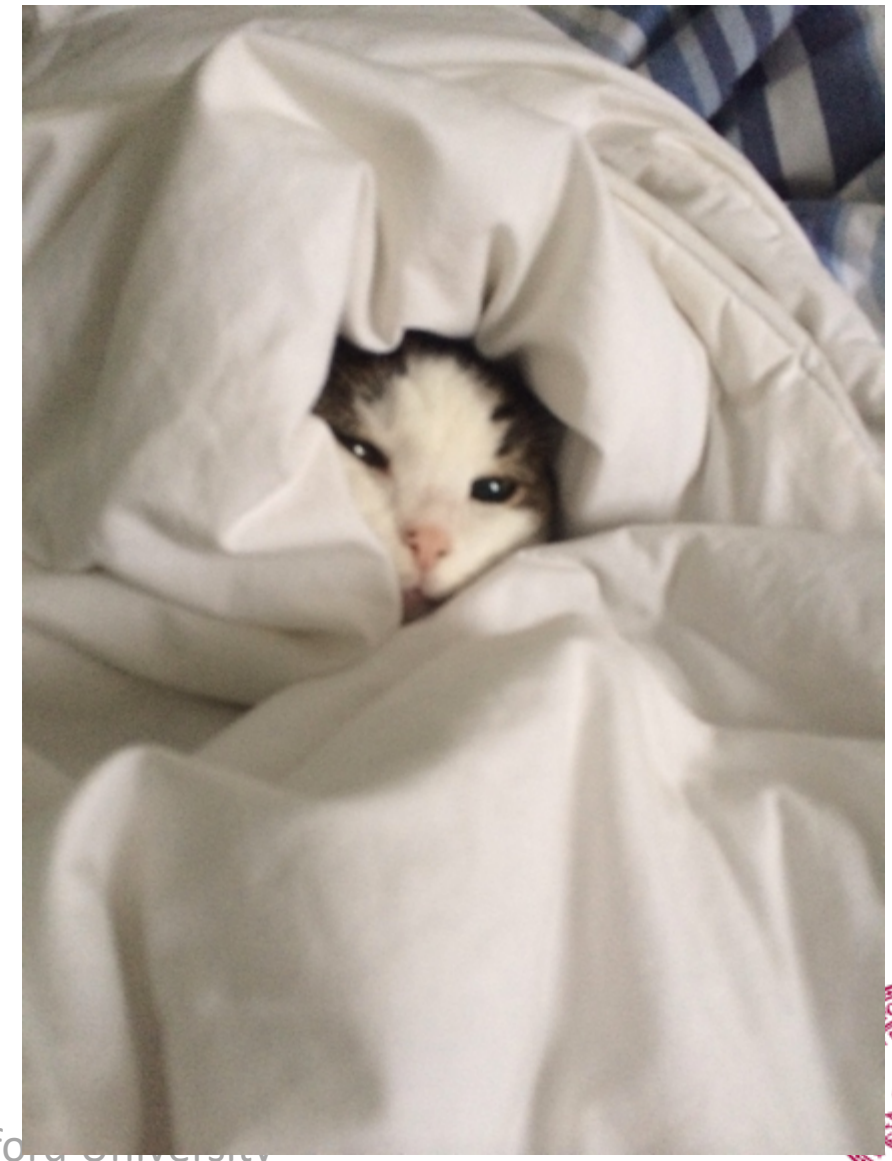
* It doesn't have to be correct all of the time

Identifying Cats

Here's one way you might code this…

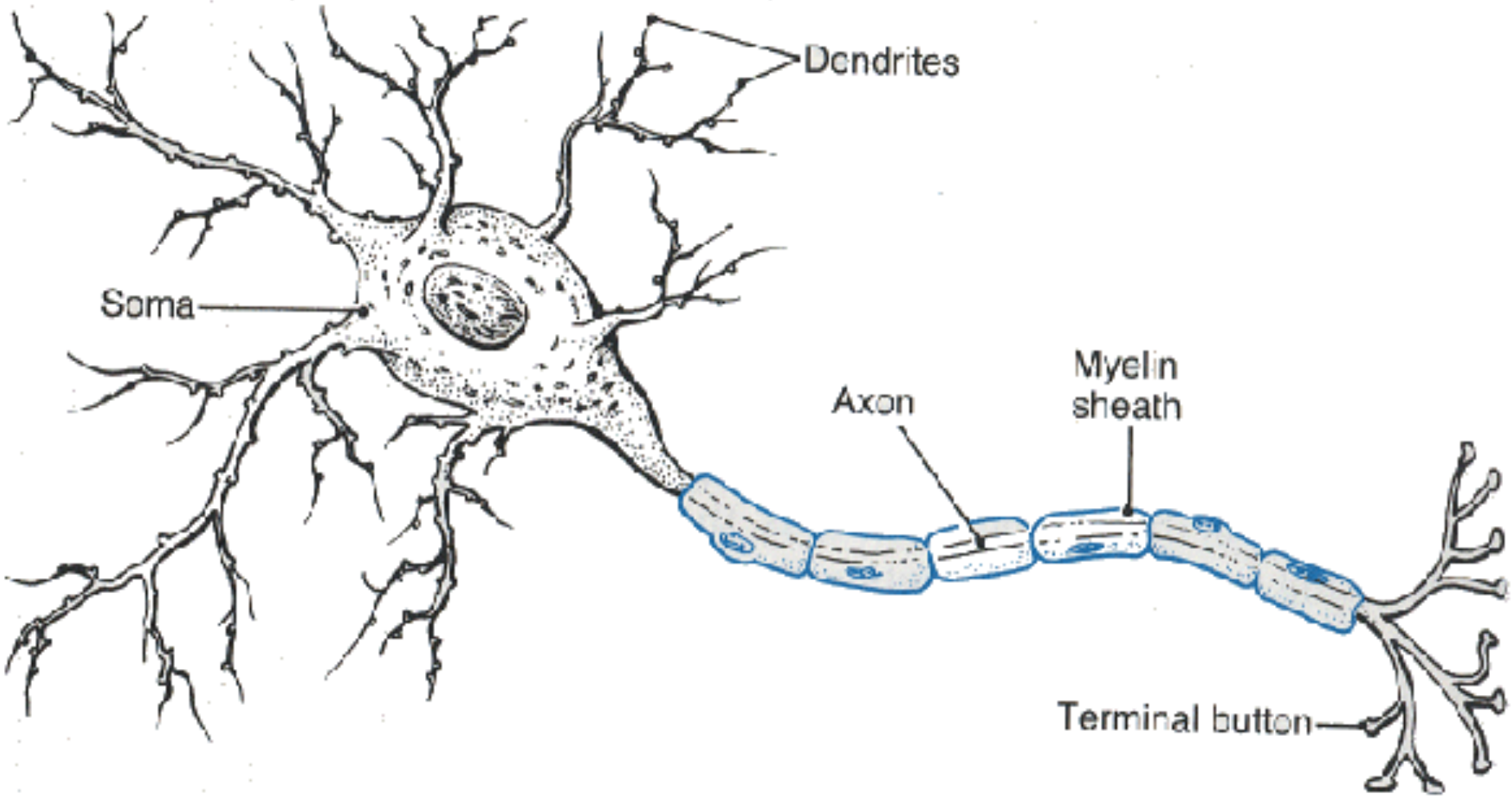
```
def is_cat(image):
    if contains_two_eyes(image):
        if has_whiskers(image):
            if has_pointy_ears(image):
                return True
    return False
```

Some Tricky Cases

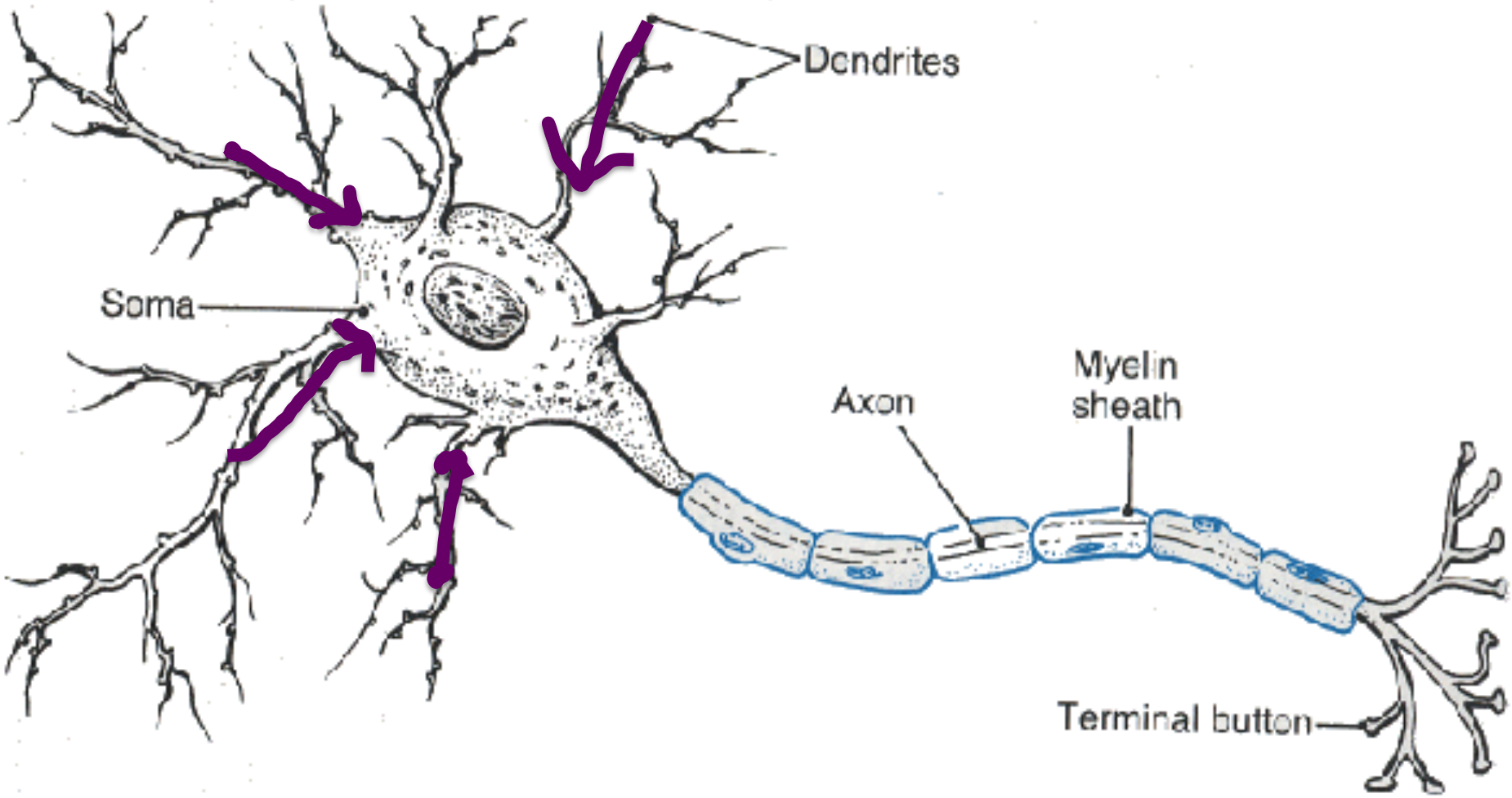


Great idea inspired by biology

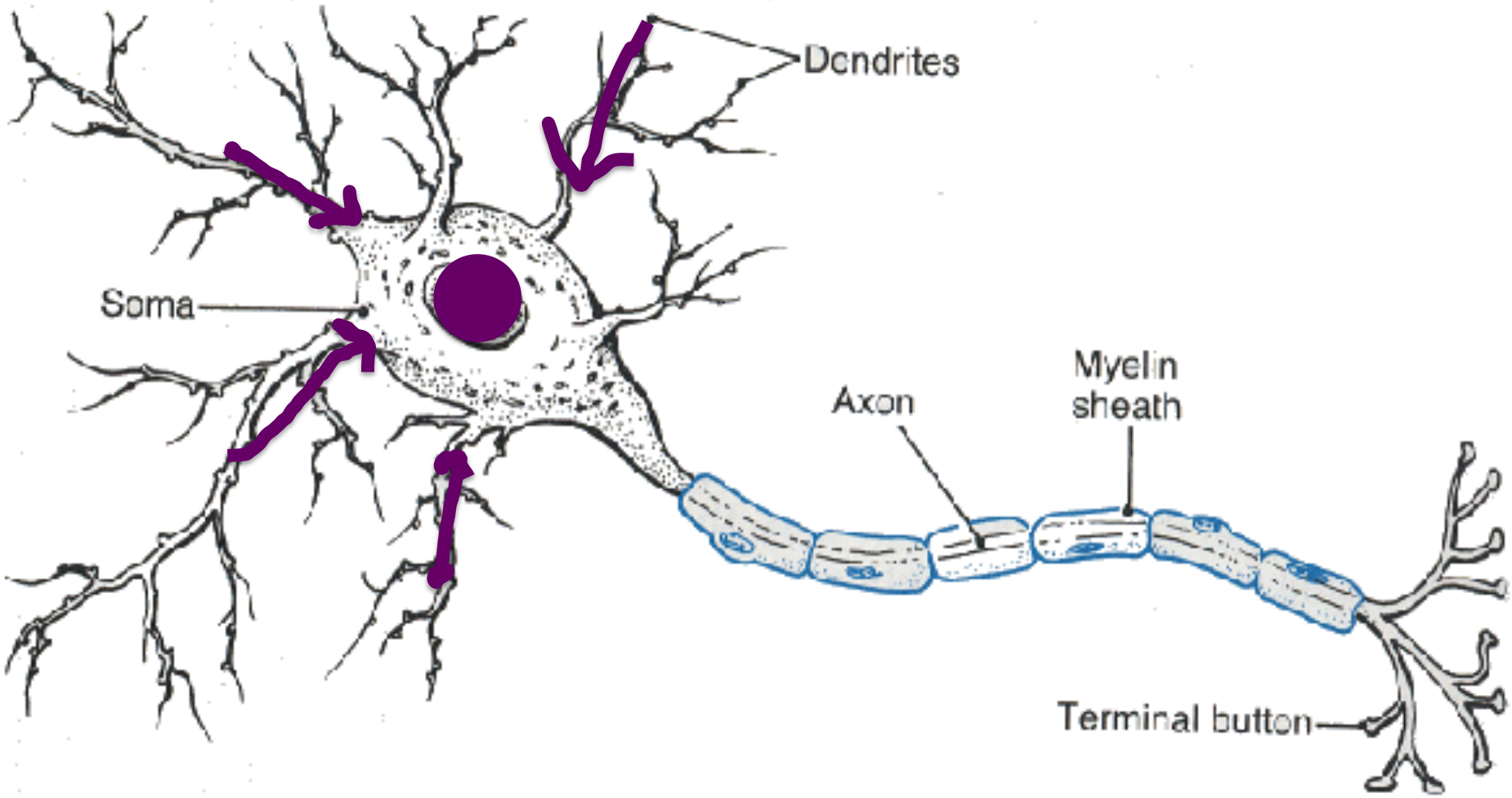
Neuron



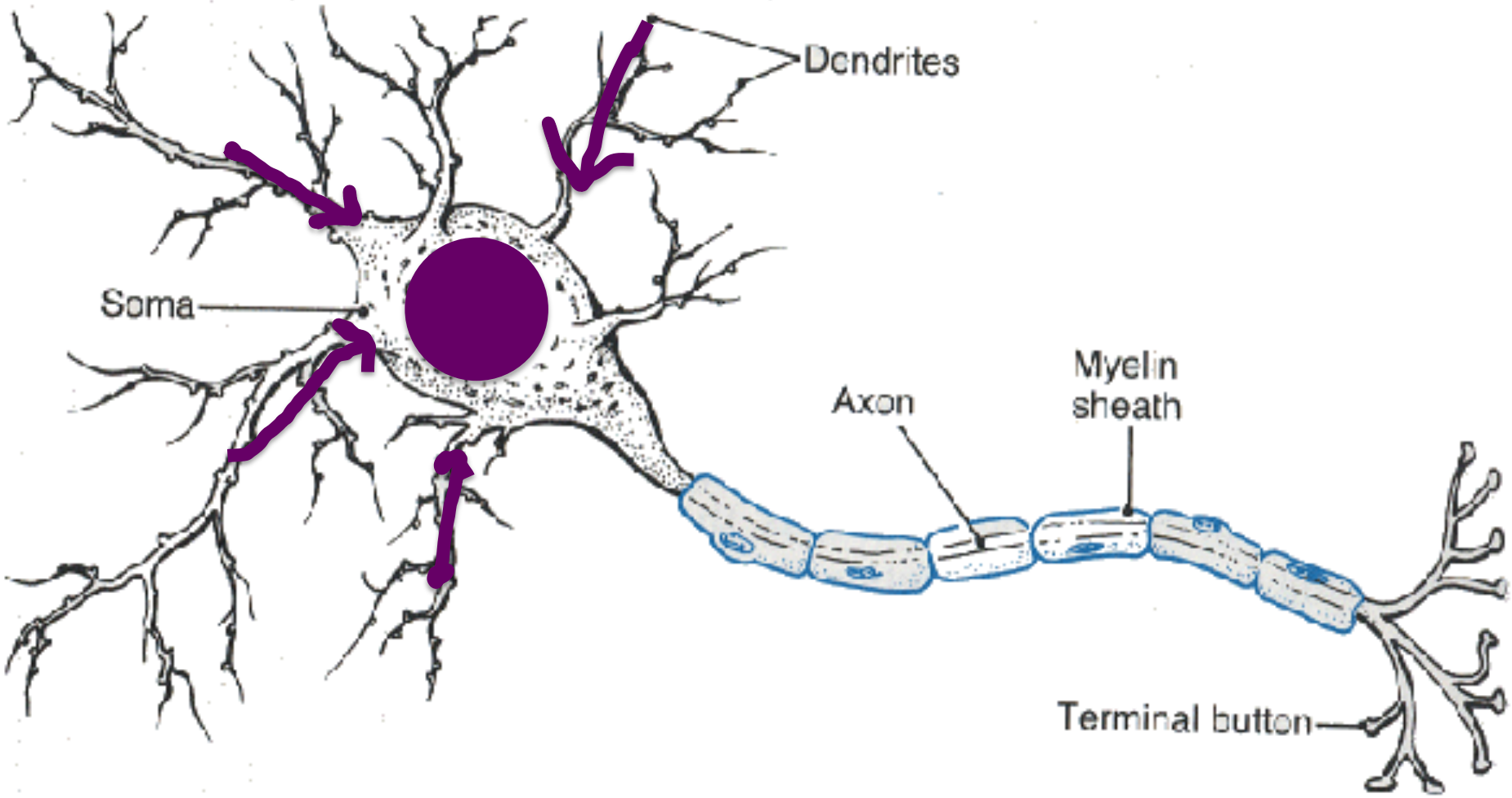
Neuron



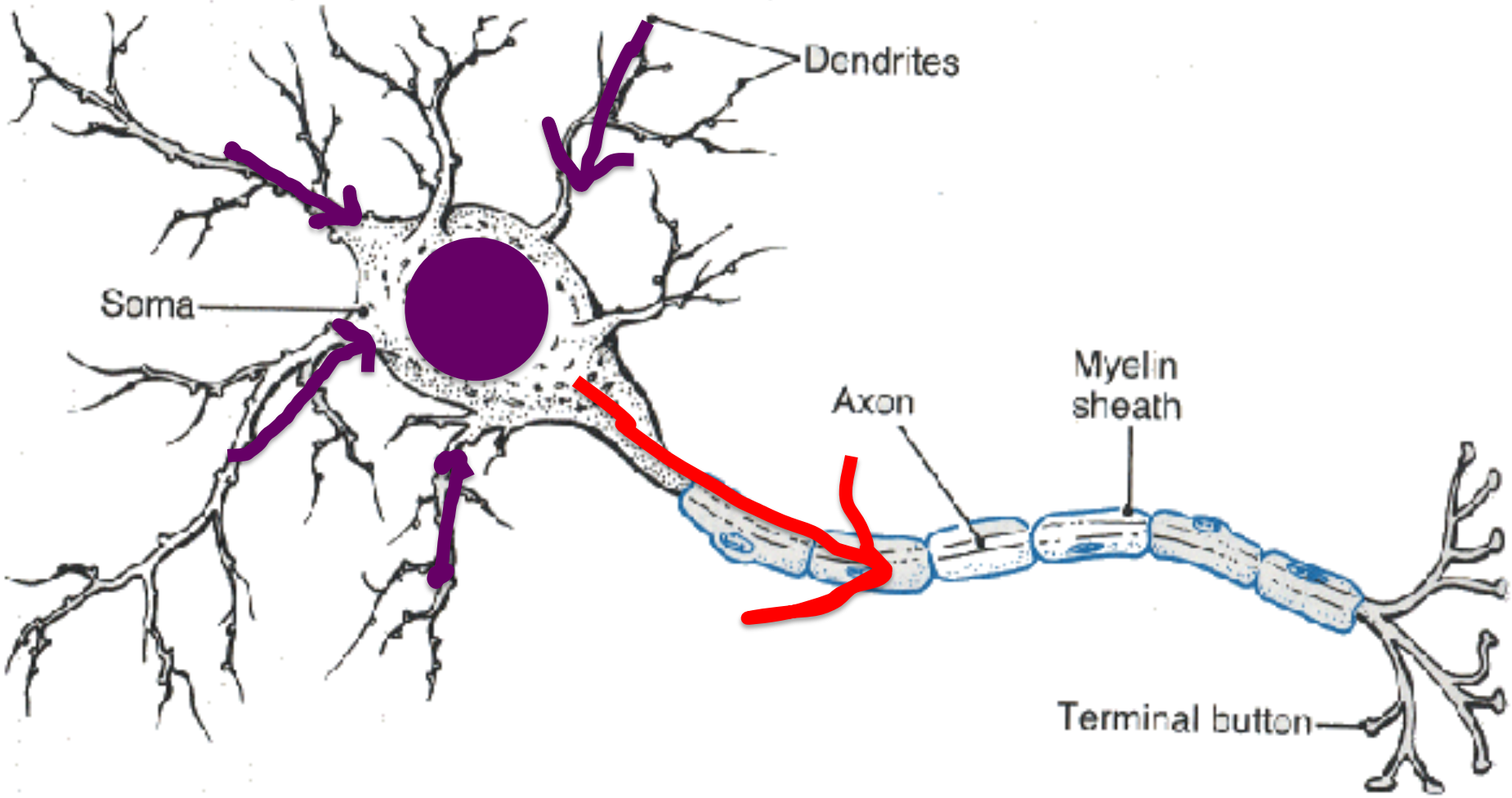
Neuron



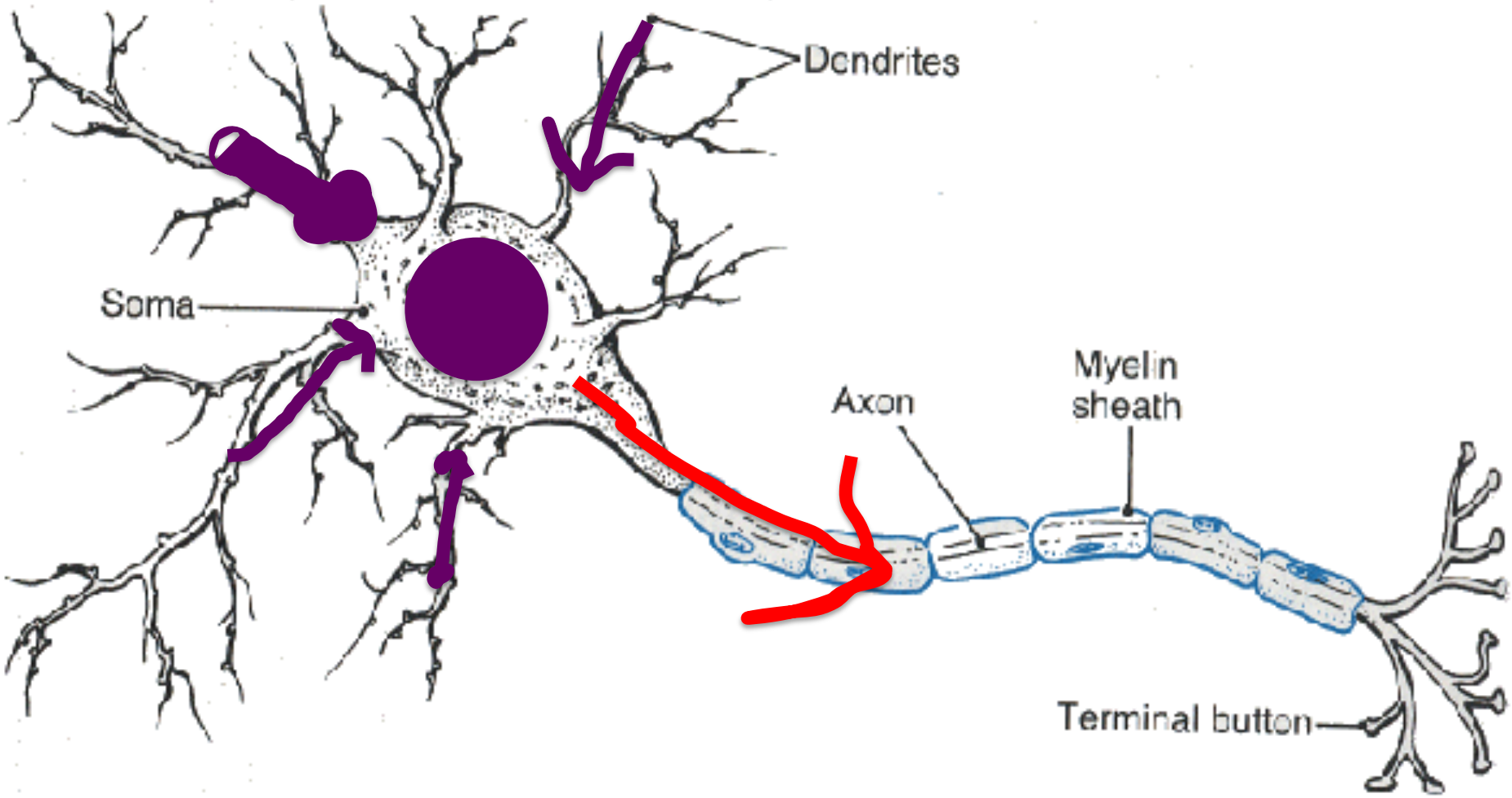
Neuron



Neuron



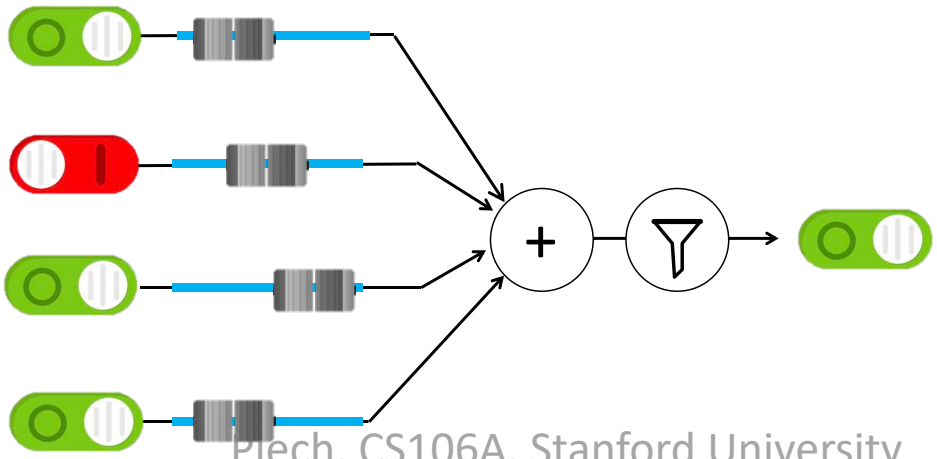
Some Inputs are More Important



Artificial Neuron

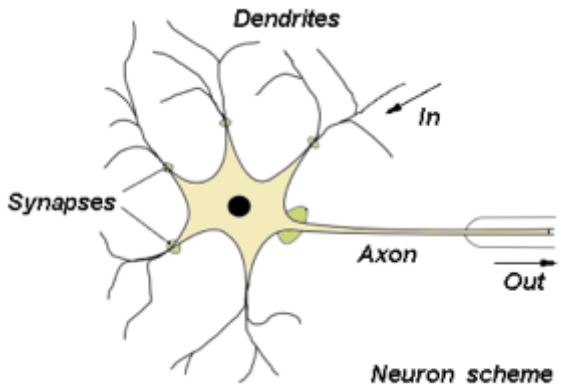
```
# calculate the activation of a neuron
def activate(weights_list, inputs_list):
    weighted_sum = 0;
    for i in range(len(inputs)):
       weighted_sum += weights_list[i] * inputs_list[i]

    return squash(weighted_sum);
```

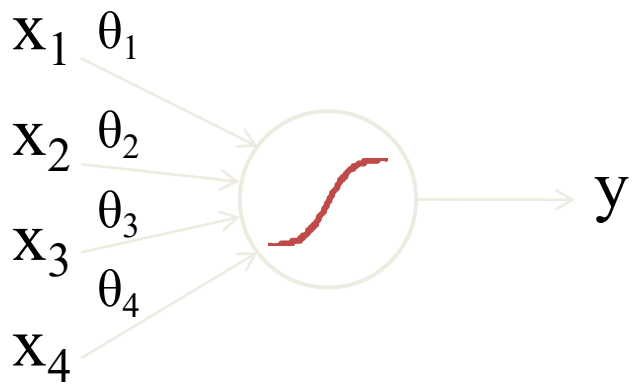
```
# the sigmoid function forces a value to be between 0 and 1
def squash(value):
    return 1 / (1 + math.exp(-value));
```



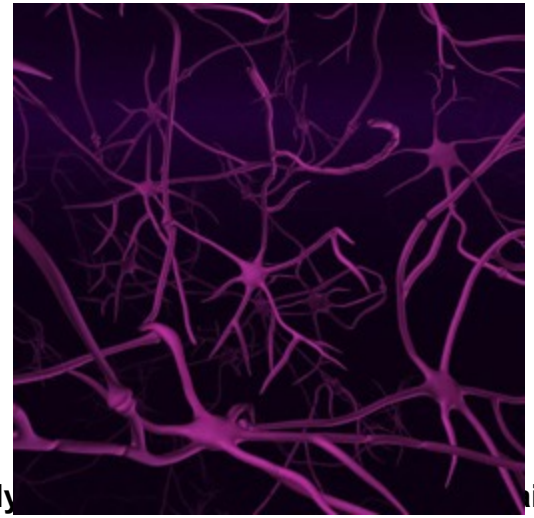
Biological Basis for Neural Networks

- A neuron

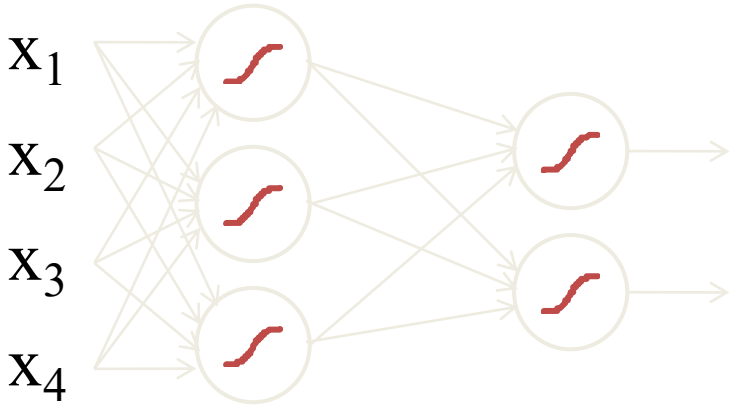




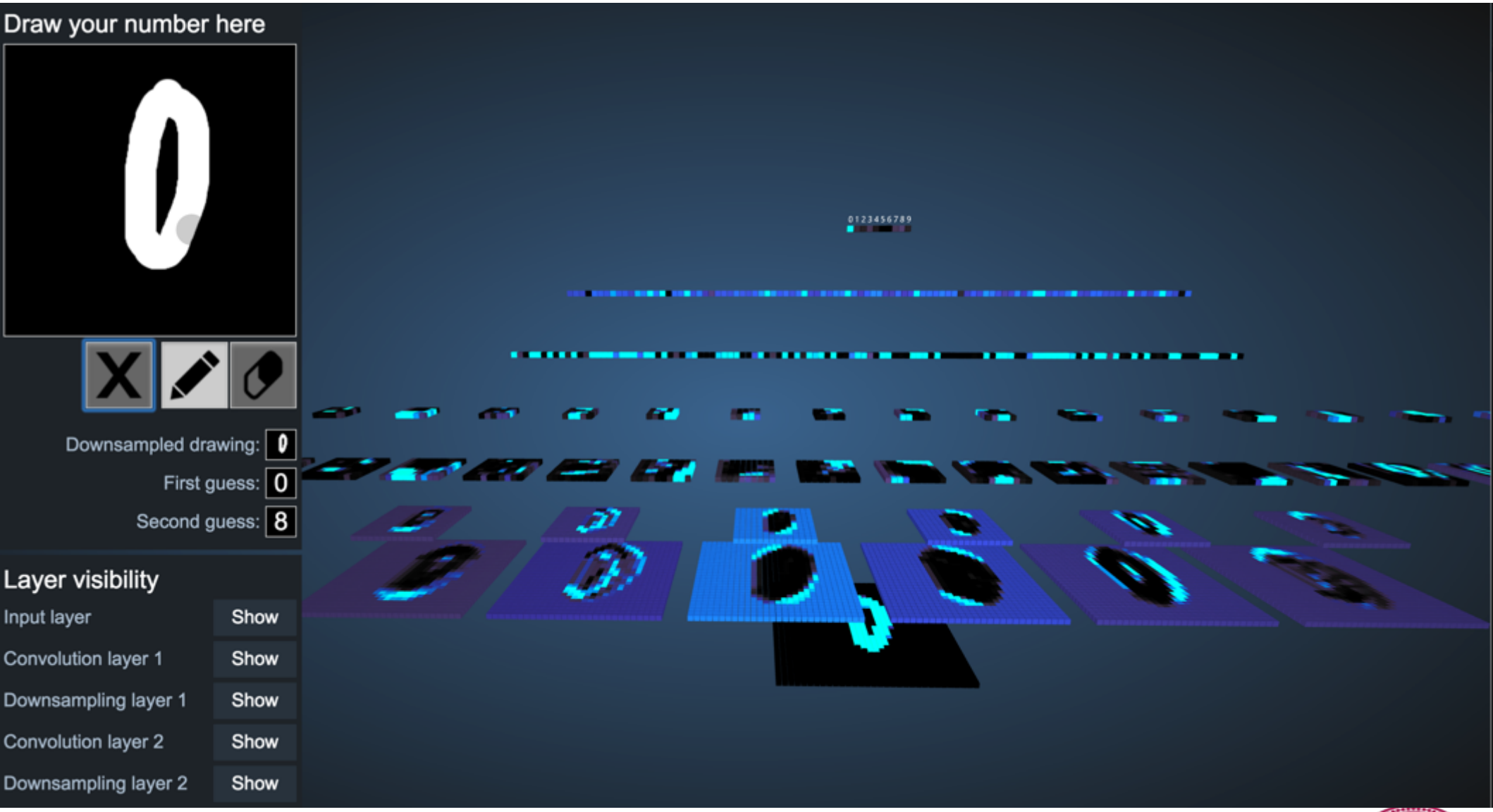
- Your brain



Actually, your brain



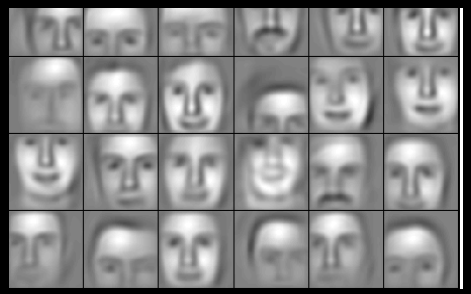
Demonstration



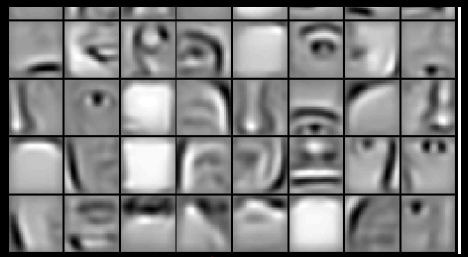
http://scs.ryerson.ca/~aharley/vis/conv/

Visualize the Weights

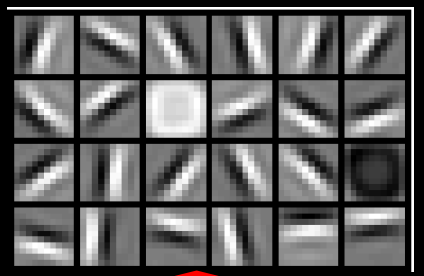
Training set: Aligned images of faces.



object models



object parts (combination of edges)

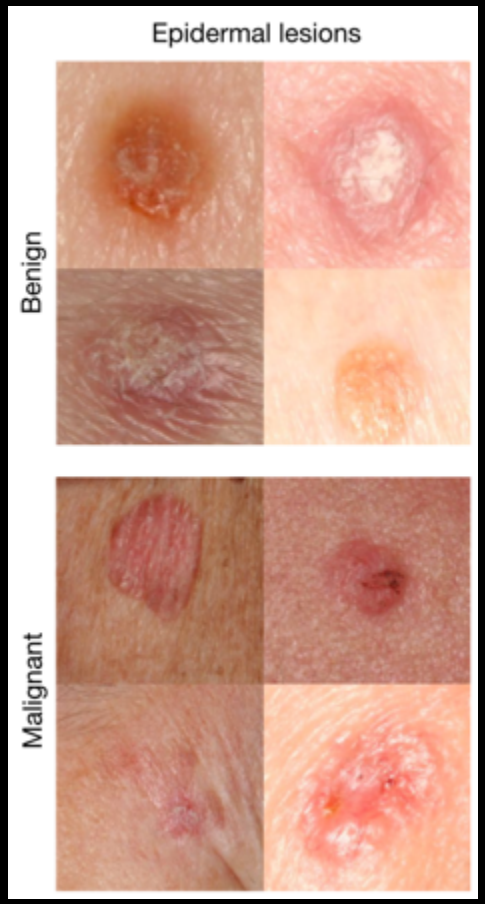


edges

pixels

[Honglak Lee]

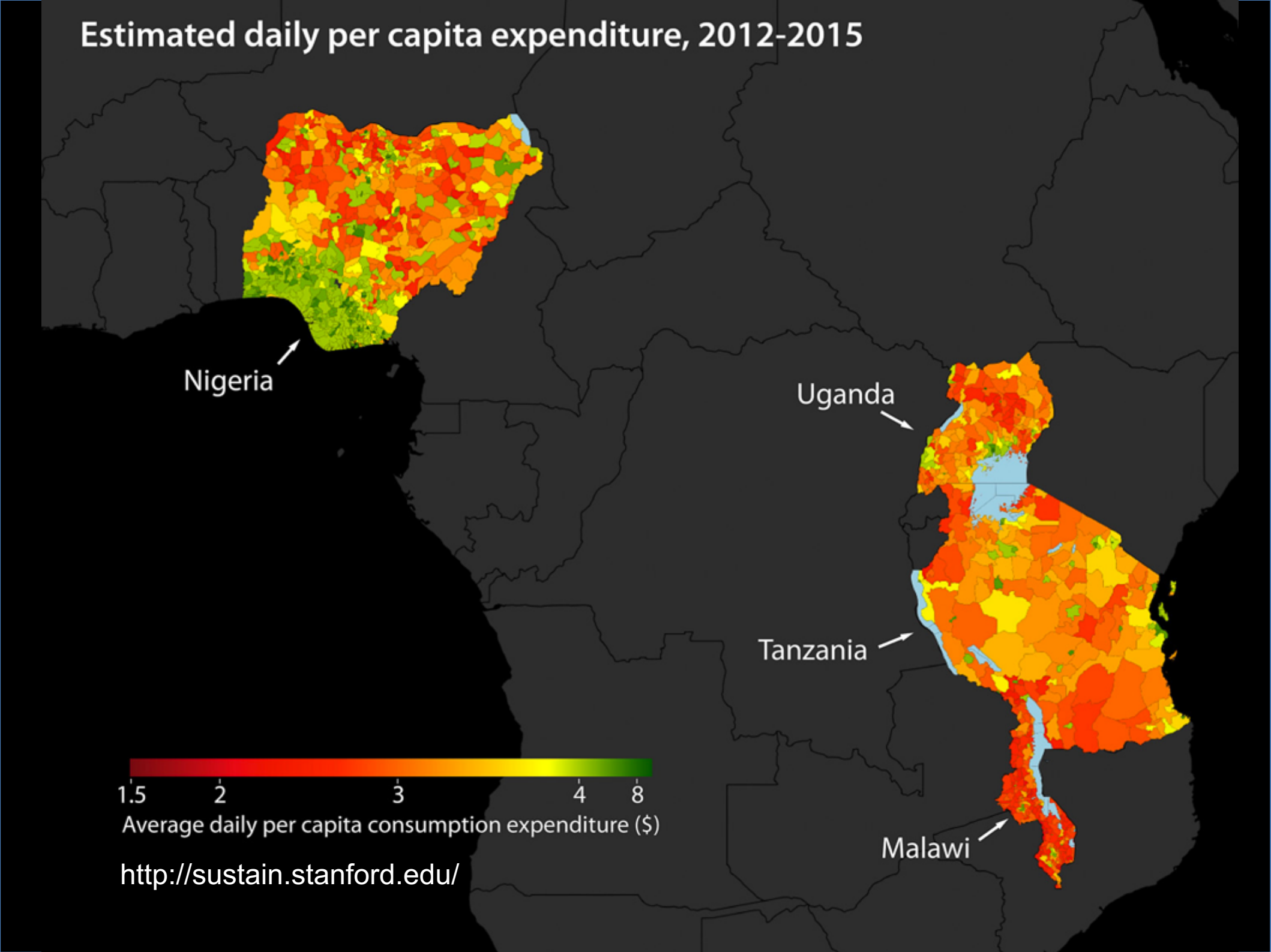
Where is this useful?



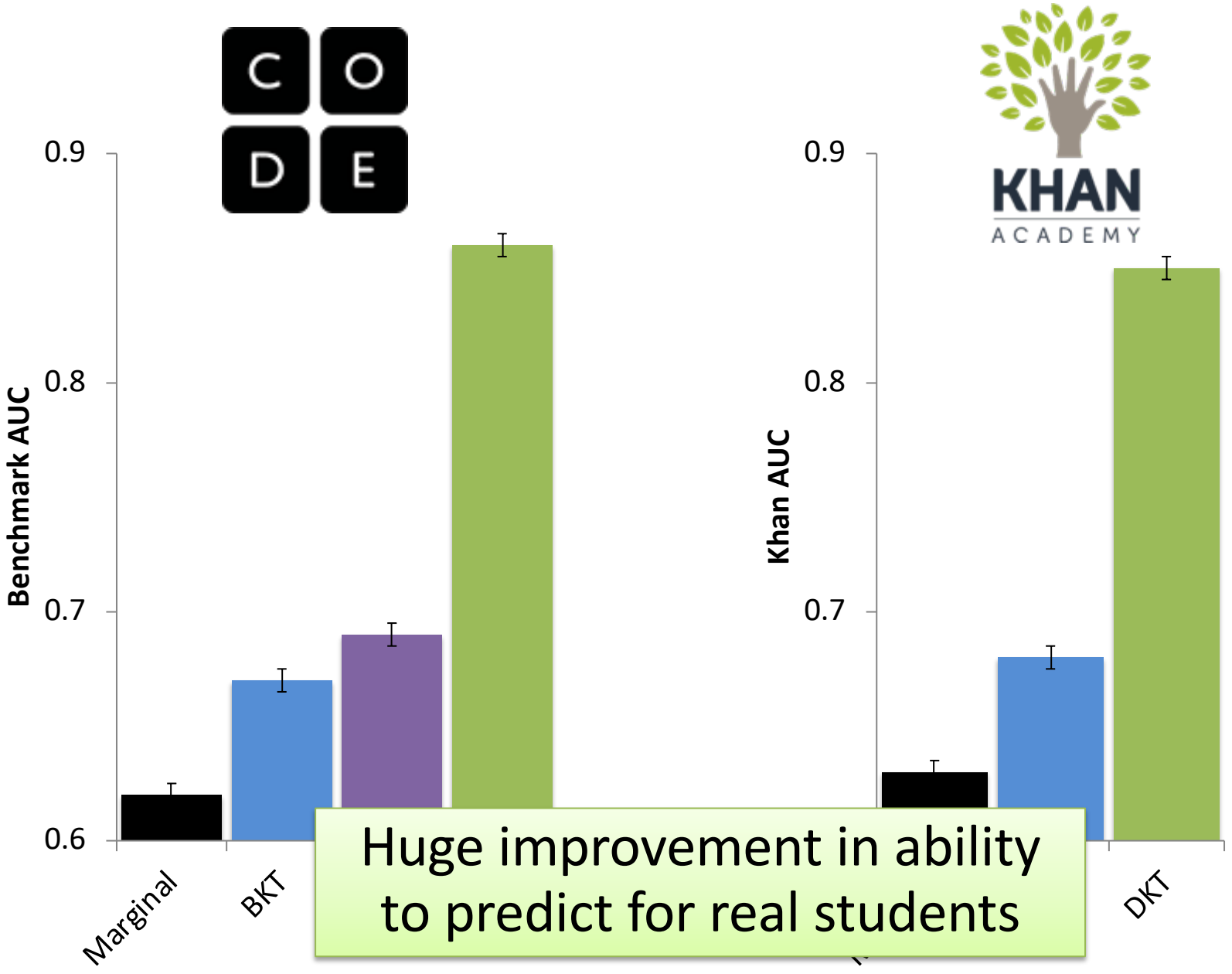
A machine learning algorithm performs **better than** the best dermatologists.

Developed this year, at Stanford.

Esteva, Andre, et al. "Dermatologist-level classification of skin cancer with deep neural networks." *Nature* 542.7639 (2017): 115-118.


Estimated daily per capita expenditure, 2012-2015
Nigeria
Uganda
Tanzania
Malawi
1.5
2
3
4
8
Average daily per capita consumption expenditure ($)
http://sustain.stanford.edu/

Understanding Students



Huge improvement in ability to predict for real students

What does it look like?

deep_learning.py

```python
import torch.nn as nn
from torch.optim import Adam
from torch.optim import SGD
import torch
import torch.nn.functional as F
import math
import pickle
import numpy as np
import csv

N_COUNTRIES = 329

def main():
  # 1. get some data (either simulate or load)
  data = load_NWEA_data()
  # 2. build a pytorch model
  model = DeepLearningModel()
  # 3. run optimization
  optimize(model, data)

# sometimes its more fun to load data...
def loadData():
  return pickle.load(open('data.pkl', 'rb'))

# Define the model you are optimizing over...
# Written as a pytorch model so we can optimize
class DeepLearningModel(nn.Module):
  # the initialize method
  def __init__(self):
    super().__init__() # necessary

    # learning rates for each of the countries
    self.theta = nn.Parameter(torch.ones(N_COUNTRIES))

    # the global parameters for the function. In the writeup
    # I call these phis (but in code I give them names so I
    # dont get confused).
    self.offset_param = nn.Parameter(torch.ones(1) * 150.0)
    self.scale_param = nn.Parameter(torch.ones(1) * 1500.0)
    self.amplitude_param = nn.Parameter(torch.ones(1) * 9.0)
    self.floor_param = nn.Parameter(torch.ones(1) * -1)

  # This is the parametric-family written in pytorch code.
  # See the writeup for details. Note how we use one-hot
  # vectors to select the theta from a student's country.
  # The input to this function is all your data as matrices
  def forward(self, alpha, country_one_hot):
    # n x 1 vector which has the country theta selected for
    # each student in the batch. (basically chose a theta from
    # the country thetas for each student)
    ability = torch.matmul(country_one_hot,self.theta).unsqueeze(1)

    # The normal distribution exponential
```