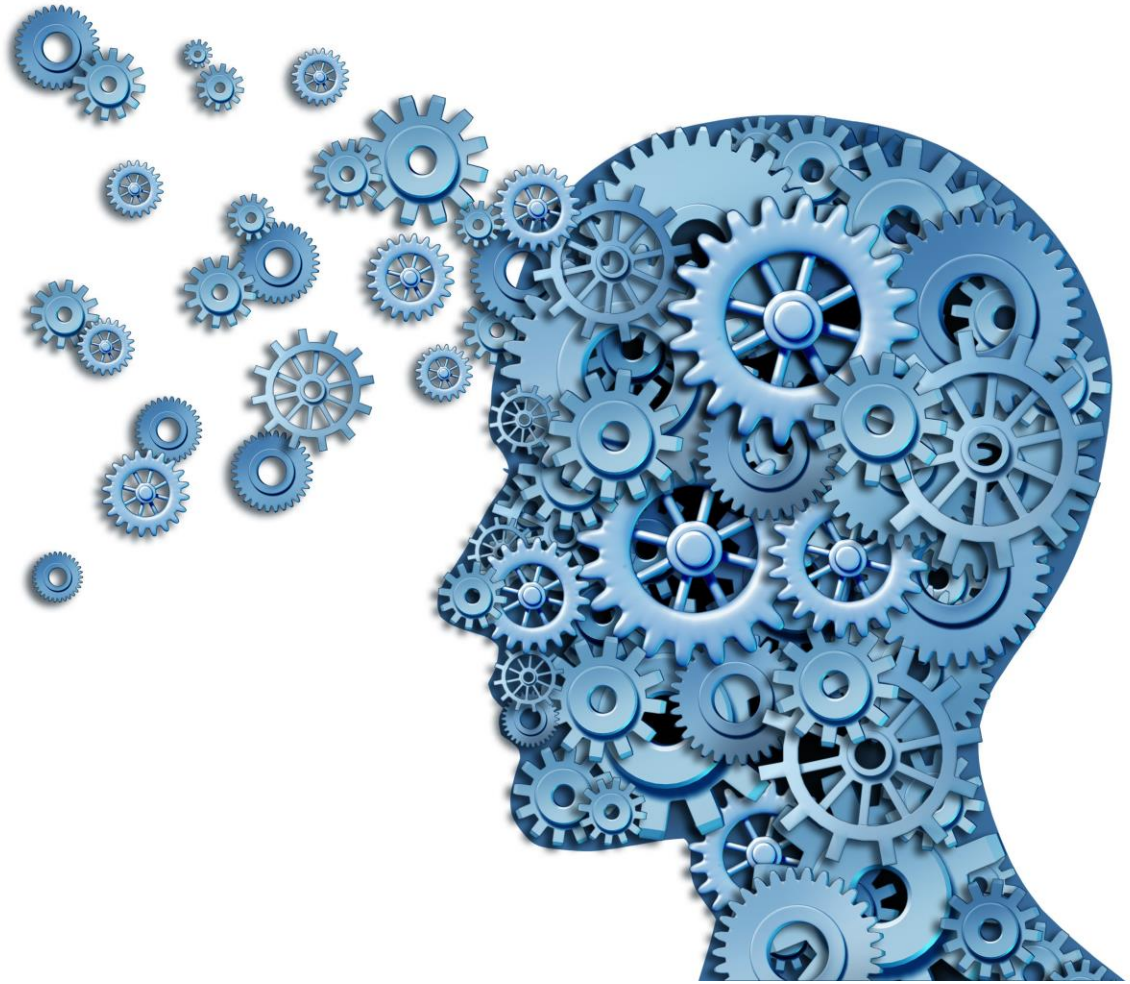
Neural & Behavioral Modeling

Week 0

Course Introduction



by Tsung-Ren (Tren) Huang 黃從仁

Course Instructor

Tsung-Ren (Tren) Huang 黃從仁, Associate Professor

Department of Psychology

Graduate Institute of Brain and Mind Sciences

Neurobiology & Cognitive Science Center

Imaging Center for Integrated Body, Mind, & Culture Research

Division of Learning Technology, Digital Learning Center

Institute of Applied Mathematical Sciences

Master Program in Statistics

Center for Artificial Intelligence and Advanced Robotics

NVIDIA-NTU Artificial Intelligence Joint Research Center

First things first

All are welcome!

You don’t need my permission to sit in.

高孟暉

Contact TAs on NTU Cool to get a registration code for enrollment.

I "may" have recordings of this course.

Philosophy and Features of Modeling

Richard Feynman: “What I cannot *build*. I do not *understand*.”

Hypothesis-driven (as opposed to data-driven), just like empirical ex periments. In fact, we are carrying out *computational experiments*.

Generative in nature, just like statistical *hypothesis-testing* or statisti cal simulations. In fact, we rely on inferential statistics to draw concl usions from the results of computer simulations.

My Teaching Goals

Expose you to the full spectrum of modeling

This allows you to understand mathematical/computational studies

Help you figure out whether this direction is for you

If not, you have also further narrowed your career search

Help you develop computational thinking

This includes a habit of asking “why” & “what if”

What models/modeling can offer

Conceptual clarity and mechanistical insights

that goes beyond phenomena

Causal inferences rather than observational correlations

that lead to policies

Falsification/support but not validation for a hypothesis/theory

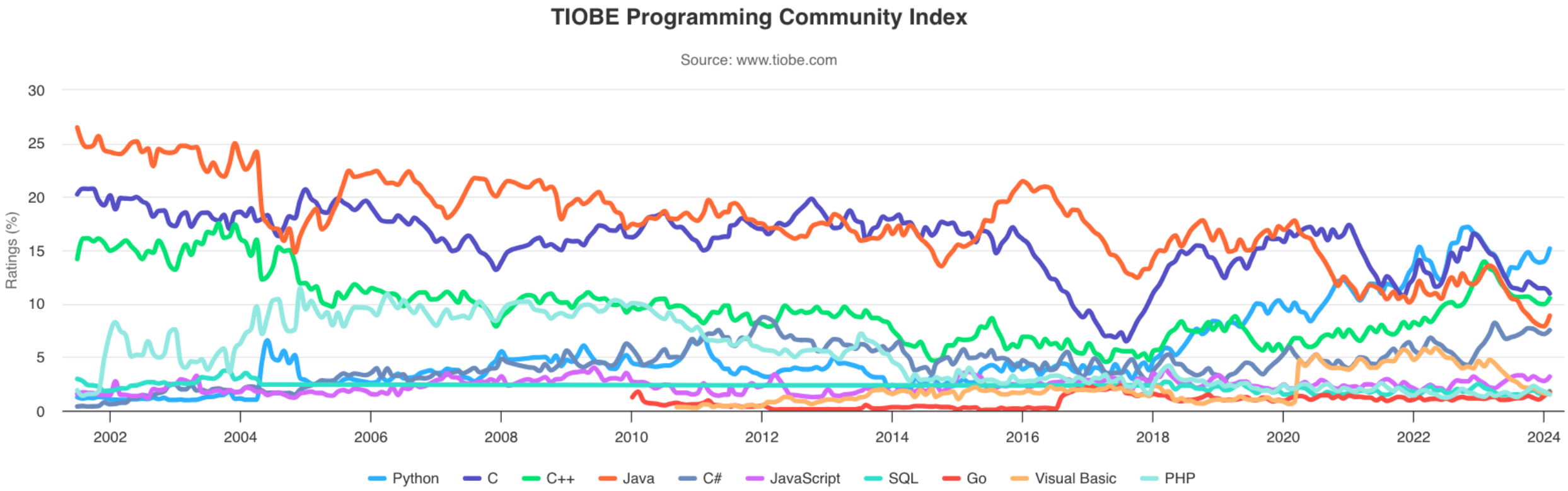
After all, **H → D** is equivalent to **¬D → ¬H** but not **D → H**

What this course offers

週次	日期	單元主題
第1週	2/21	Course Introduction: Models & modeling
第2週	2/28	和平紀念日(放假)
第3週	3/6	Behavioral Modeling (1/2): System dynamics
第4週	3/13	Behavioral Modeling (2/2): Agent-based modeling
第5週	3/20	Computational Cognitive Science (1/2): Basics
第6週	3/27	Computational Cognitive Science (2/2): Advanced
第7週	4/3	Computational Cognitive Neuroscience (1/3): Modeling principles & canonical neural computation
第8週	4/10	Computational Cognitive Neuroscience (2/3): Neural Networks
第9週	4/17	Computational Cognitive Neuroscience (3/3): Learning & Memory
第10週	4/24	Deep-learning Neural Networks (1/4): Fully-Connected Multilayer Perceptron (MLP)
第11週	5/1	Deep-learning Neural Networks (2/4): Convolutional Neural Network (CNN)
第12週	5/8	Deep-learning Neural Networks (3/4): Recurrent Neural Networks (RNN)
第13週	5/15	Deep-learning Neural Networks (4/4): Deep Reinforcement Learning (RL) & Advanced Networks
第14週	5/22	Computational Neuroscience (1/2): 1 spiking neuron
第15週	5/29	Computational Neuroscience (2/2): N spiking neuron
第16週	6/5	無期末考/課程

Programming Languages

As of February 2024, Python ranks first in terms of popularity



Python Environments

Personal Computing:

Recommend VS Code + Extensions (Python + Jupyter)

Private Cloud:

Recommend Jupyter Notebook in Anaconda Python 3

Public Cloud:

Recommend Google Colab, especially for AI & deep learning

Copilot with ChatGPT

VS Code:

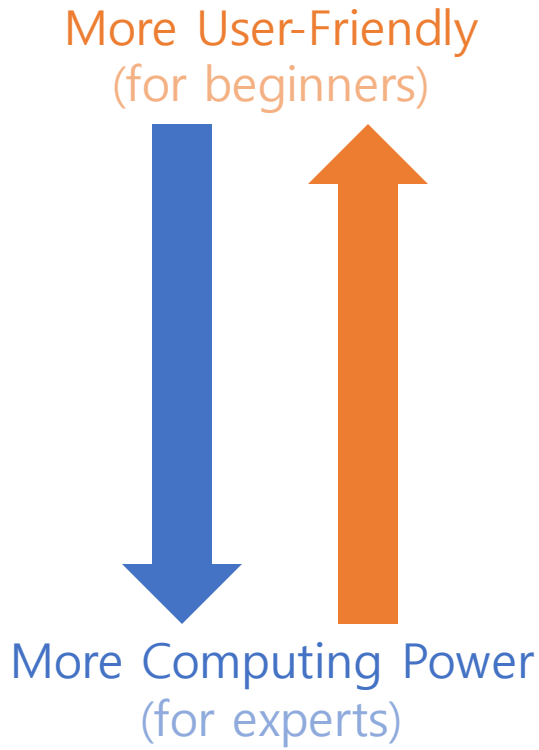
Install these extensions: GitHub Copilot + ChatGPT

Jyputer:

Install this Chrome/Edge/Brave extension

Colab:

Install this Chrome/Edge/Brave extension



VS Code/Jupyter/Colab

Use [Shift] + [Enter] for execution

Remember this at least

Use [Tab] for auto-completion (not for Colab)

Or for exploration of functions under a library

Use [Shift] + [Tab] for help (not for VS Code & Colab)

Holding [Shift] + more [Tab] to get more info

PYTHON FOR DATA SCIENCE CHEAT SHEET

Python Basics

Datatypes

- Numbers: a=2(Integer), b=2.0(Float), c=1+2j(Complex)
- List: a=[1,2,3,'Word']
- Tuple: a= (1,2,4)
- String: a="New String"
- Sets: a= {2,3,4,5}
- Dictionary: x= {'a': [1,2],'b': [4,6]}

Operators

Numeric Operator: (Say, a holds 5, b holds 10)

- a + b = 15
- a – b = -5
- a * b = 50
- 7.0//2.0 = 3.0, -11//3 = -4
- b/a = 2
- b % a = 0
- a**b =9765625

Comparison Operator:

- (a == b): not true
- (a!= b): true
- (a > b): not true
- (a > b): not true
- (a >= b): not true
- (a <= b) is true

Boolean Operator:

- a and b
- a or b
- not a

Operations

List Operations

- List=[]: Defines an empty list
- list[i]=a: Stores a at the ith position
- list[i]: Retrieves the character at the ith position
- list[i:j]: Retrieves characters in the range i to j
- list.append(val): Adds item at the end
- list.pop([i]): Removes and returns item at index i

String Operations

- String[i]: Retrieves the character at the ith position
- String[i:j]: Retrieves characters in the range i to j

Dictionary Operations

- dict={} : Defines an empty dictionary
- dict[i]=a: stores "a" to the key "i"
- dict[i]: Retrieves the item with the key "i"
- dict.key: Gives all the key items
- dict.values: Gives all the values

OOPS

Inheritance:

A process of using details from a new class without modifying existing class.

Polymorphism:

A concept of using common operation in different ways for different data input.

Encapsulation:

Hiding the private details of a class from other objects.

Class/object

Class:

```
class Pen:
    pass
```

Object: obj=Pen()

Flow Control Method

- **if-else** (Conditional Statement)

```
if price>=700:
        print("Buy.")
else:
        print("Don't buy.")
```

- **For loop** (Iterative Loop Statement)

```
a="New Text"
count=0
for i in a:
    if i=='e':
        count=count+1
print(count)
```

- **While loop** (Conditional Loop Statement)

```
a=0
i=1
while i <10:
        a=a*2
        i=i+1
print(a)
```

- **Loop Control:** Break, Pass and continue

Functions

```
def new_function():
    print("Hello World")

new_function()
```

Lambda Function

```
lambda a,b: a+b

lambda a,b: a*b
```

Comments

```
# Single Line Comment
"""
Multi-line comment
"""
```

Generic Operations

- range(5): 0,1,2,3,4
- S=input("Enter:")
- Len(a): Gives item count in a
- min(a): Gives minimum value in a
- max(a): Gives minimum value in a
- sum(a): Adds up items of an iterable and returns sum
- sorted(a): Sorted list copy of a
- importing modules: import random

File Operations

f= open("File Name","opening mode")

(Opening modes: r: read, w: write, a: append, r+: both read and write)

Try & Except Block

```
try:
        [Statement body block]
        raise Exception()
except Exception as e:
        [Error processing block]
```

FURTHERMORE:

Python for Data Science Certification Training Course

Python For Data Science *Cheat Sheet*

Python Basics

Learn More Python for Data Science Interactively at www.datacamp.com

Variables and Data Types

Variable Assignment

```
>>> x=5
>>> x
 5
```

Calculations With Variables

```
>>> x+2
 7
>>> x-2
 3
>>> x*2
 10
>>> x**2
 25
>>> x%2
 1
>>> x/float(2)
 2.5
```

Sum of two variables
Subtraction of two variables
Multiplication of two variables
Exponentiation of a variable
Remainder of a variable
Division of a variable

Types and Type Conversion

`str()`	`'5', '3.45', 'True'`	Variables to strings
`int()`	`5, 3, 1`	Variables to integers
`float()`	`5.0, 1.0`	Variables to floats
`bool()`	`True, True, True`	Variables to booleans

Asking For Help

```
>>> help(str)
```

Lists

Also see NumPy Arrays

```
>>> a = 'is'
>>> b = 'nice'
>>> my_list = ['my', 'list', a, b]
>>> my_list2 = [[4,5,6,7], [3,4,5,6]]
```

Selecting List Elements

Index starts at 0

Subset

```
>>> my_list[1]
>>> my_list[-3]
```

Select item at index 1
Select 3rd last item

Slice

```
>>> my_list[1:3]
>>> my_list[1:]
>>> my_list[:3]
>>> my_list[:]
```

Select items at index 1 and 2
Select items after index 0
Select items before index 3
Copy my_list

Subset Lists of Lists

```
>>> my_list2[1][0]
>>> my_list2[1][:2]
```

my_list[list][itemOfList]

List Operations

```
>>> my_list + my_list
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list * 2
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
>>> my_list2 > 4
True
```

List Methods

```
>>> my_list.index(a)
>>> my_list.count(a)
>>> my_list.append('!')
>>> my_list.remove('!')
>>> del(my_list[0:1])
>>> my_list.reverse()
>>> my_list.extend('!')
>>> my_list.pop(-1)
>>> my_list.insert(0,'!')
>>> my_list.sort()
```

Get the index of an item
Count an item
Append an item at a time
Remove an item
Remove an item
Reverse the list
Append an item
Remove an item
Insert an item
Sort the list

Strings

```
>>> my_string = 'thisStringIsAwesome'
>>> my_string
'thisStringIsAwesome'
```

String Operations

```
>>> my_string * 2
 'thisStringIsAwesomethisStringIsAwesome'
>>> my_string + 'Innit'
 'thisStringIsAwesomeInnit'
>>> 'm' in my_string
 True
```

String Operations

Index starts at 0

```
>>> my_string[3]
>>> my_string[4:9]
```

String Methods

```
>>> my_string.upper()
>>> my_string.lower()
>>> my_string.count('w')
>>> my_string.replace('e', 'i')
>>> my_string.strip()
```

String to uppercase
String to lowercase
Count String elements
Replace String elements
Strip whitespaces

Libraries

Import libraries

```
>>> import numpy
>>> import numpy as np
```

Selective import

```
>>> from math import pi
```

Data analysis
Machine learning
Scientific computing
2D plotting

Install Python

ANACONDA
Leading open data science platform powered by Python

spyder
Free IDE that is included with Anaconda

jupyter
Create and share documents with live code, visualizations, text, ...

Numpy Arrays

Also see Lists

```
>>> my_list = [1, 2, 3, 4]
>>> my_array = np.array(my_list)
>>> my_2darray = np.array([[1,2,3],[4,5,6]])
```

Selecting Numpy Array Elements

Index starts at 0

Subset

```
>>> my_array[1]
  2
```

Select item at index 1

Slice

```
>>> my_array[0:2]
  array([1, 2])
```

Select items at index 0 and 1

Subset 2D Numpy arrays

```
>>> my_2darray[:,0]
  array([1, 4])
```

my_2darray[rows, columns]

Numpy Array Operations

```
>>> my_array > 3
 array([False, False, False,  True], dtype=bool)
>>> my_array * 2
 array([2, 4, 6, 8])
>>> my_array + np.array([5, 6, 7, 8])
 array([6, 8, 10, 12])
```

Numpy Array Functions

```
>>> my_array.shape
>>> np.append(other_array)
>>> np.insert(my_array, 1, 5)
>>> np.delete(my_array,[1])
>>> np.mean(my_array)
>>> np.median(my_array)
>>> my_array.corrcoef()
>>> np.std(my_array)
```

Get the dimensions of the array
Append items to an array
Insert items in an array
Delete items in an array
Mean of the array
Median of the array
Correlation coefficient
Standard deviation

CHATGPT

You

Creaet a tutorial on Python basics with codes

ChatGPT

Certainly! Here's a tutorial on Python basics that covers fundamental concepts along with code examples:

Installing Python:

Visit the official Python website (python.org) and download the latest version of Python.

Follow the installation instructions for your operating system.

Running Python Code:

Open a text editor or an Integrated Development Environment (IDE) like PyCharm, Visual Studio Code, or IDLE.

Write your Python code in a new file with a .py extension.

Save the file.

Open the terminal or command prompt, navigate to the directory where your code file is saved, and run the command `python filename.py` to execute the code.

Print Statements:

Print statements are used to display output in Python.

Use the `print()` function to print text, variables, or expressions.

Example:

```python
print("Hello, World!")
```

Variables and Data Types:

Variables are used to store data.

Python supports various data types such as integers, floating-point numbers, strings, booleans, lists, tuples, and dictionaries.

Example:

```python
age = 25
name = "John"
is_student = True
```

User Input:

Use the `input()` function to prompt the user for input.

The `input()` function returns a string, so you may need to convert it to the appropriate data type.

Example:

```python
```

Enter to send, Shift+Enter for a new line, / to choose prompt

test.ipynb

test.ipynb > a=1

+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ··· Python 3.9.6

```
print("Hello, World!")
```

[2] ✓ 0.0s Python

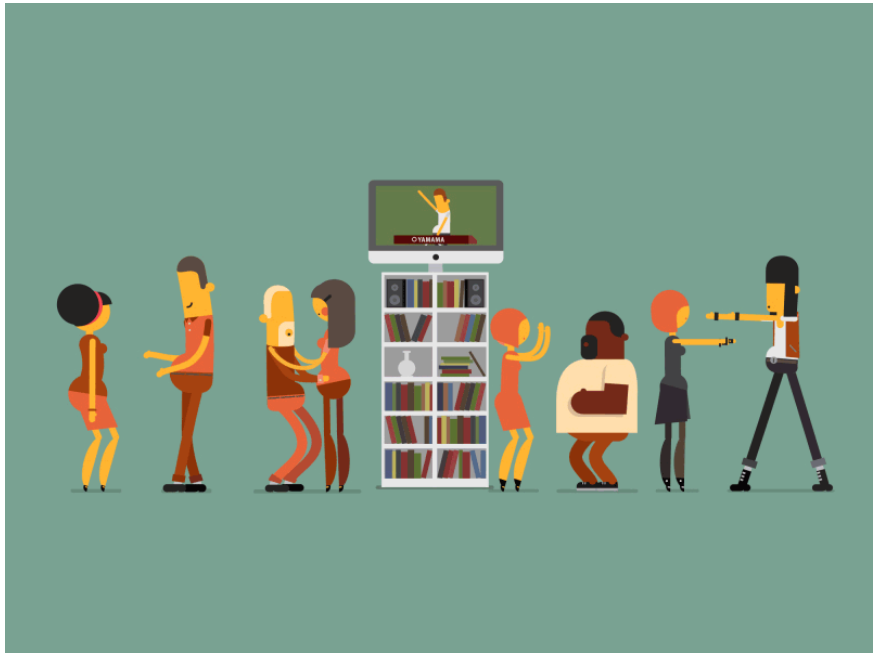
··· 4

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER Tasks

Ln 1, Col 23 Spaces: 4 LF

Demo: Party simulation

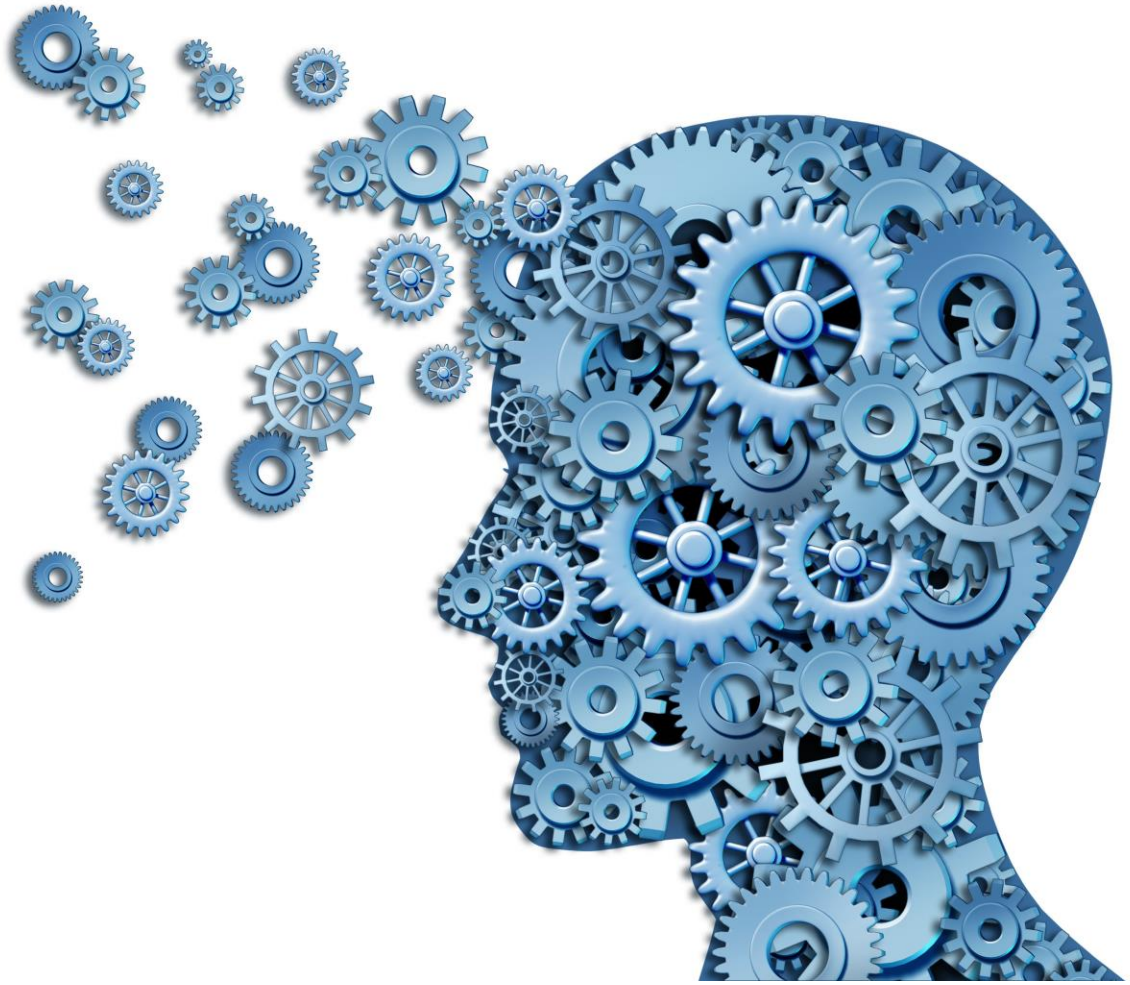
What will happen if one cannot tolerate others of the opposite sex?



What if one can somewhat tolerate others of the opposite sex?

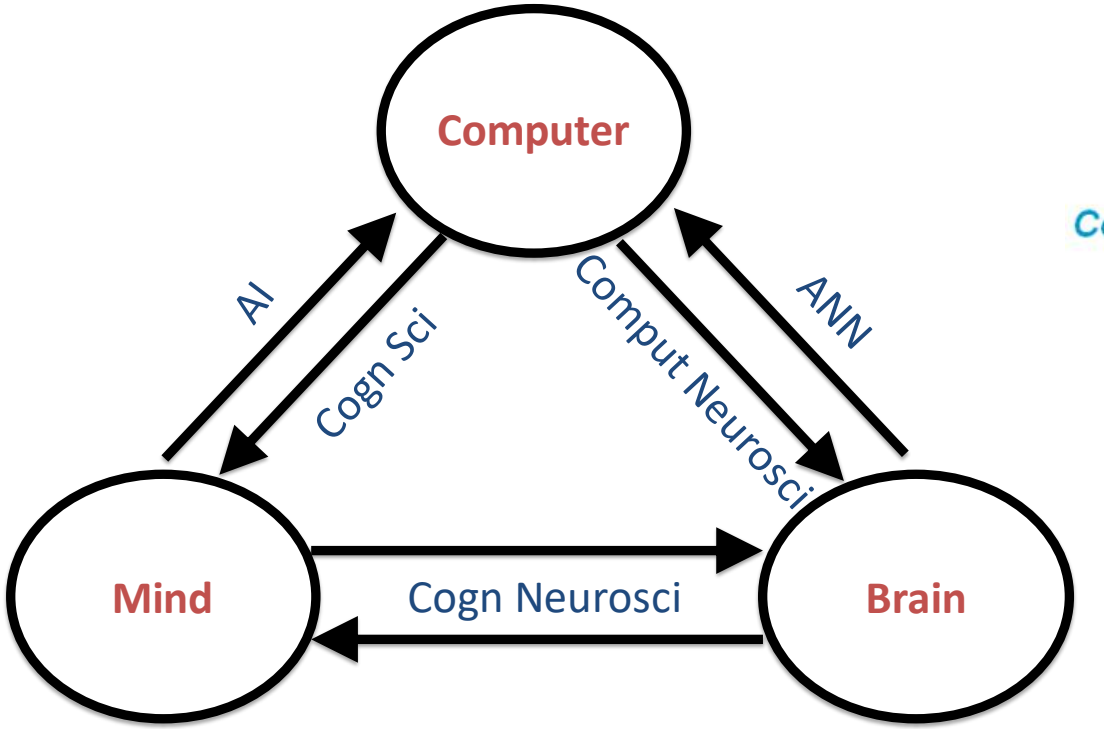
Neural & Behavioral Modeling

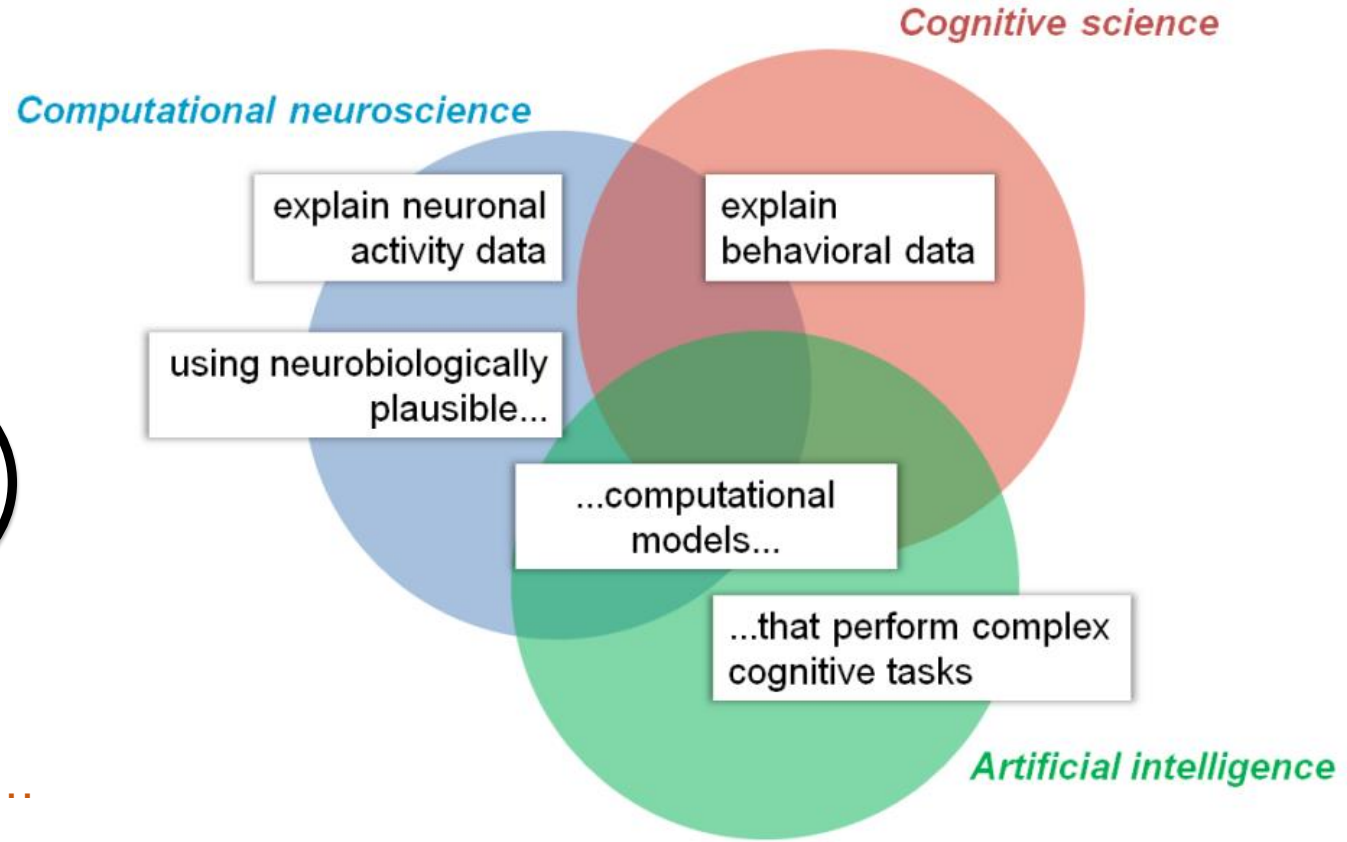
Week 1
Background



by Tsung-Ren (Tren) Huang 黃從仁

We are studying at the intersections



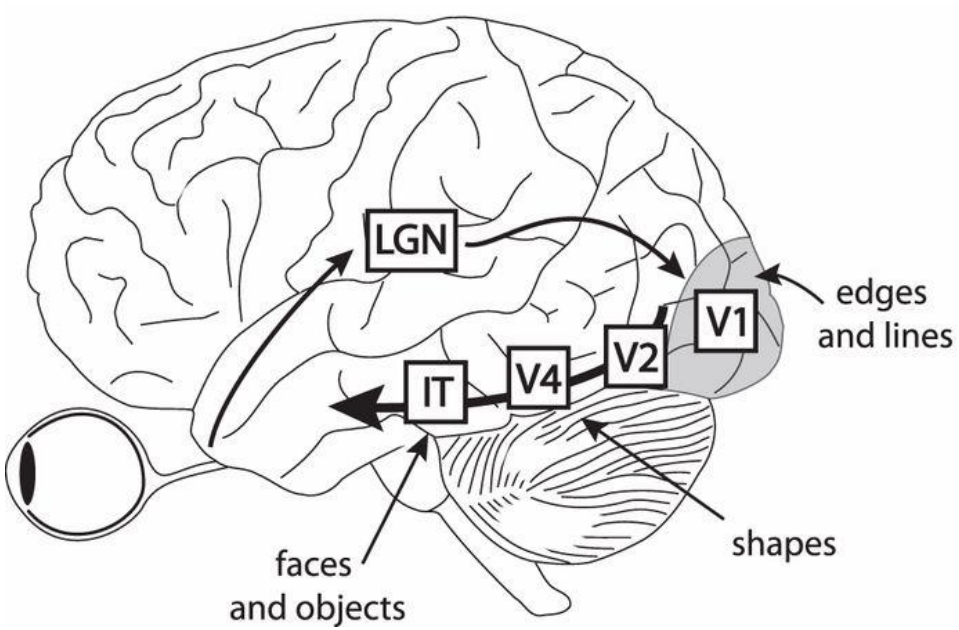


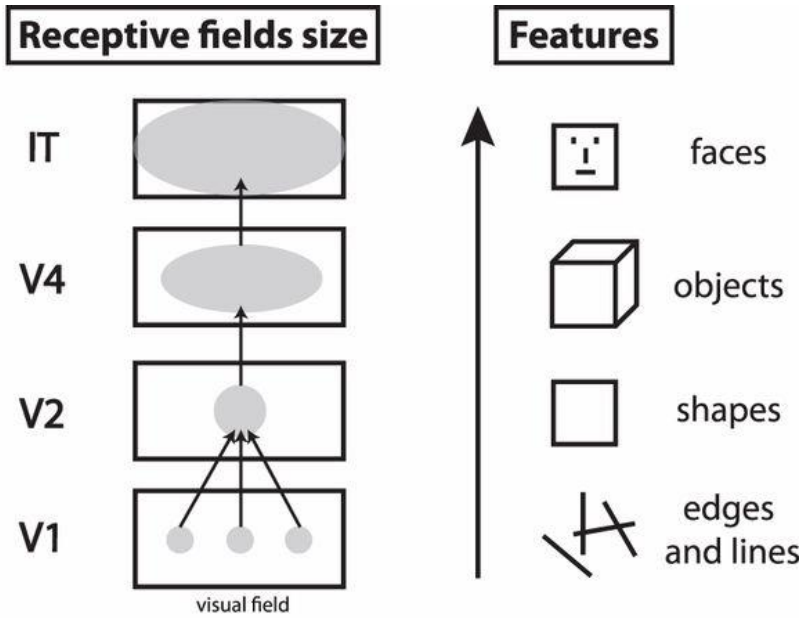
Relevant Journals:

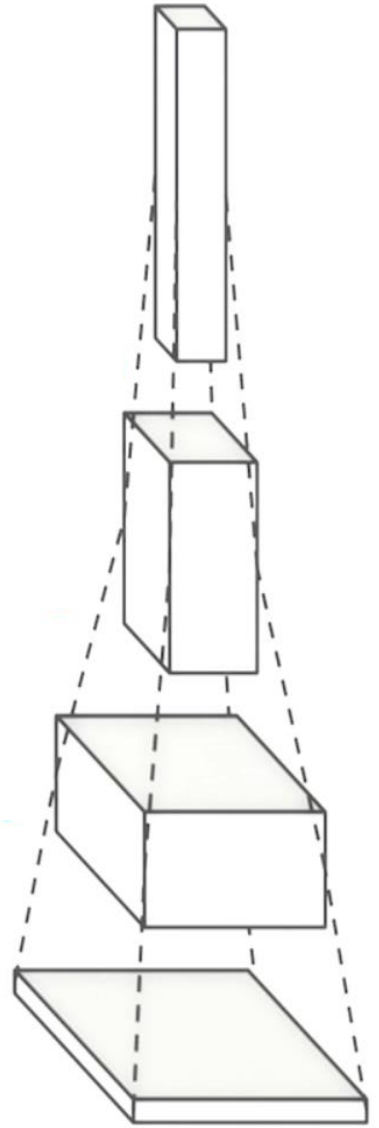
- **Psychology**: Psychological Review (IF=8.934) …
- **Neuroscience**: Journal of Cognitive Neuroscience (IF=3.225) …
- **AI**: IEEE Transactions on Affective Computing (IF=11.2) …
- **Robotics**: International Journal of Social Robotics (IF=4.7) …

Biological inspirations: AI design

CNN emulates the visual ventral pathway

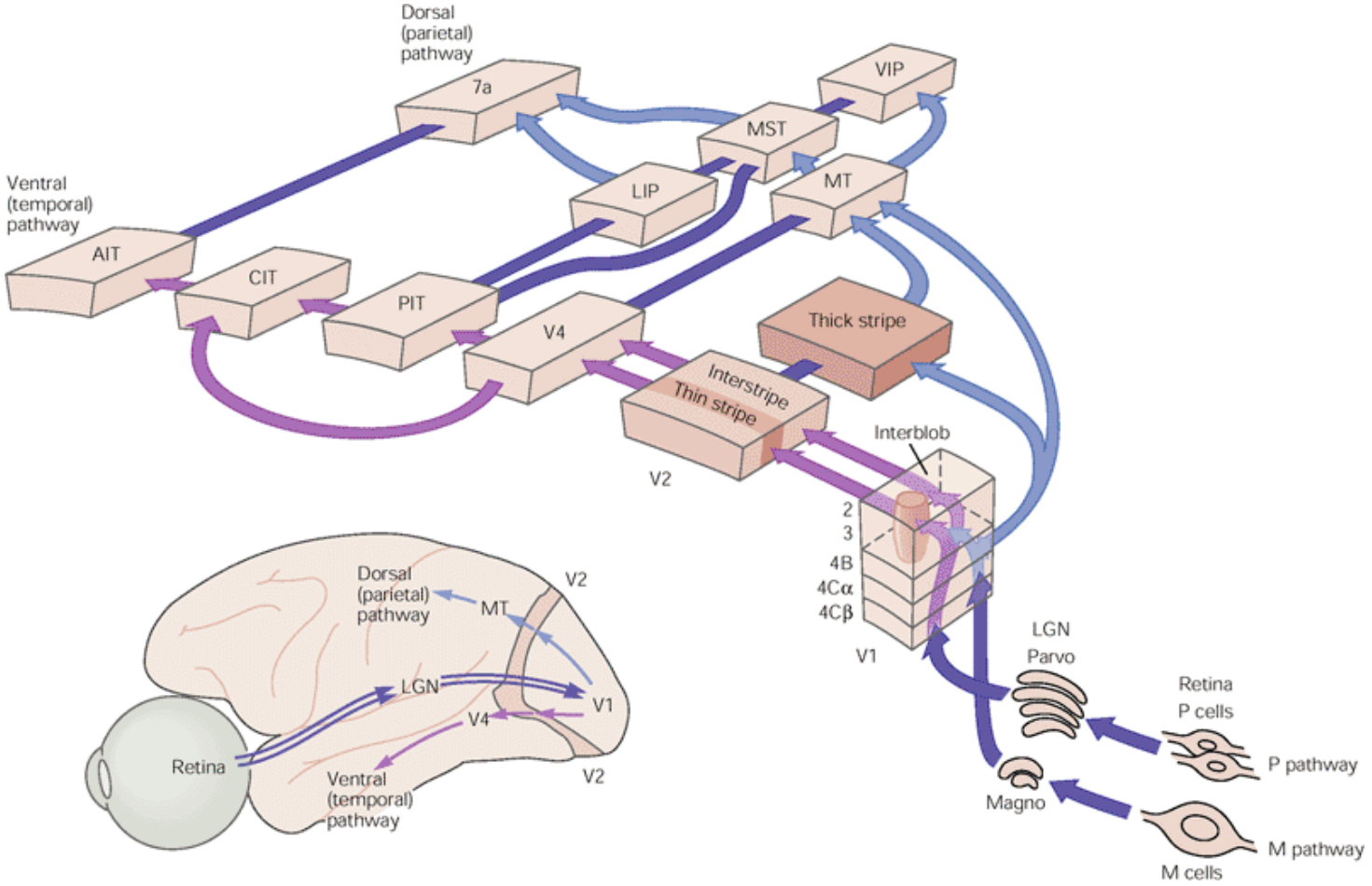






Biological inspirations: AI design

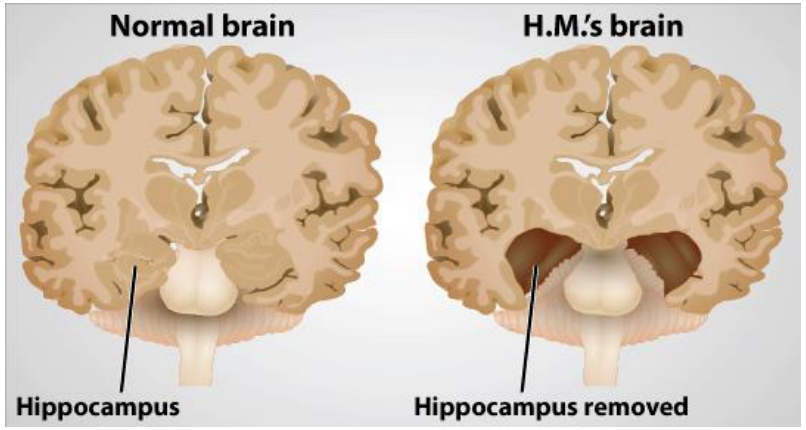
Ventral pathway is insufficient for more complex vision problem.



Dorsal pathway is also needed!

Biological inspirations: Explainable AI

Like neuroanatomy, lesions help understand a neural system

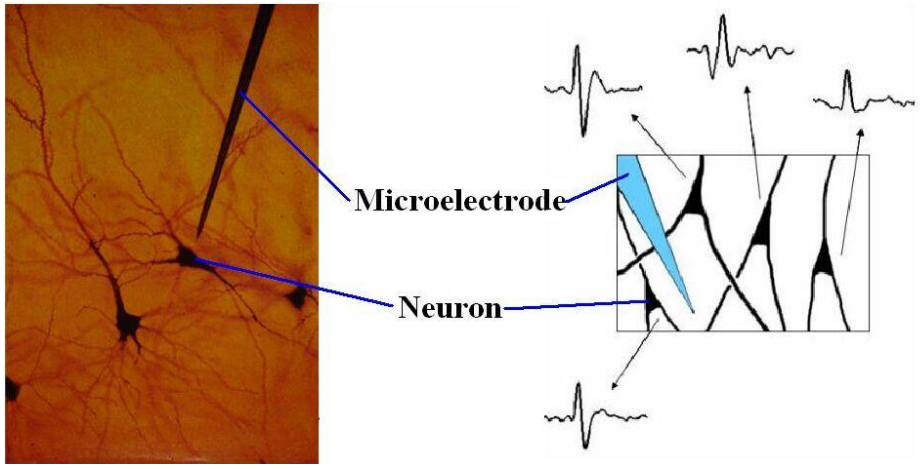


Error %	Train Top-1	Val Top-1	Val Top-5
Our replication of Krizhevsky *et al.* [18], 1 convnet	35.1	40.5	18.1
Removed layers 3,4	41.8	45.4	22.1
Removed layer 7	27.4	40.0	18.4
Removed layers 6,7	27.4	44.8	22.4
Removed layer 3,4,6,7	71.1	71.3	50.1
Adjust layers 6,7: 2048 units	40.3	41.7	18.8
Adjust layers 6,7: 8192 units	26.8	40.0	18.1

Zeiler & Fergus, 201
4

Biological inspirations: Explainable AI

Like neurophysiology, activity recordings reveal neuronal tunings



Karpathy et al., 2016, ICL

R

Cell sensitive to position in line:

```
The sole importance of the crossing of the Berezina lies in the fact
that it plainly and indubitably proved the fallacy of all the plans for
cutting off the enemy's retreat and the soundness of the only possible
line of action--the one Kutuzov and the general mass of the army
demanded--namely, simply to follow the enemy up. The French crowd fled
at a continually increasing speed and all its energy was directed to
reaching its goal. It fled like a wounded animal and it was impossible
to block its path. This was shown not so much by the arrangements it
made for crossing as by what took place at the bridges. When the bridges
broke down, unarmed soldiers, people from Moscow and women with children
who were with the French transport, all--carried on by vis inertiae--
pressed forward into boats and into the ice-covered water and did not,
surrender.
```

Cell that turns on inside quotes:

```
"You mean to imply that I have nothing to eat out of.... On the
contrary, I can supply you with everything even if you want to give
dinner parties," warmly replied Chichagov, who tried by every word he
spoke to prove his own rectitude and therefore imagined Kutuzov to be
animated by the same desire.

Kutuzov, shrugging his shoulders, replied with his subtle penetrating
smile: "I meant merely to say what I said."
```

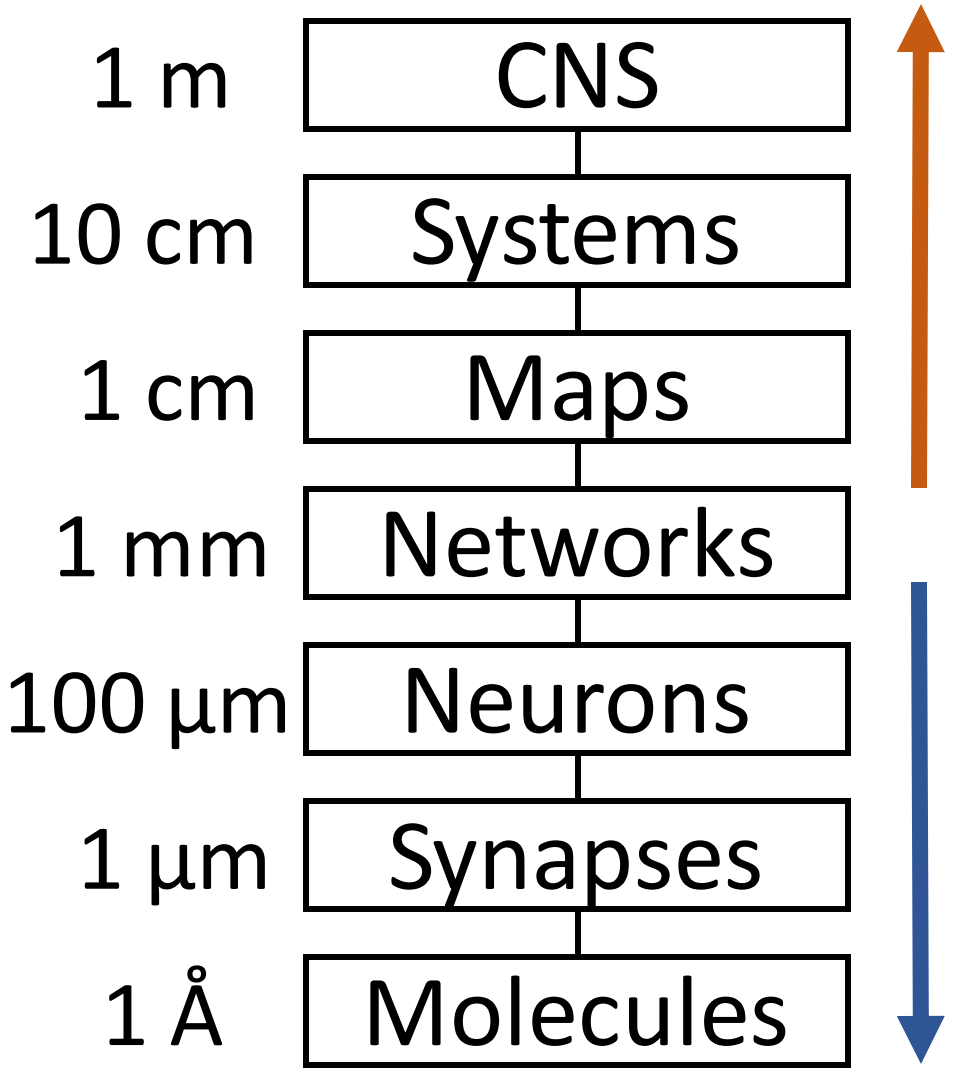
Cell that robustly activates inside if statements:

```
static int __dequeue_signal(struct sigpending *pending, sigset_t *mask,
        siginfo_t *info)
{
  int sig = next_signal(pending, mask);
  if (sig) {
    if (current->notifier) {
      if (sigismember(current->notifier_mask, sig)) {
        if (!(current->notifier)(current->notifier_data)) {
          clear_thread_flag(TIF_SIGPENDING);
          return 0;
        }
      }
    }
    collect_signal(sig, pending, info);
  }
  return sig;
}
```

A large portion of cells are not easily interpretable. Here is a typical example:

```
/* Unpack a filter field's string representation from user-space
 * buffer. */
char *audit_unpack_string(void **bufp, size_t *remain, size_t len)
{
  char *str;
  if (!*bufp || (len == 0) || (len > *remain))
    return ERR_PTR(-EINVAL);
  /* Of the currently implemented string fields, PATH_MAX
   * defines the longest valid length.
   */
```

Taxonomy based on spatial scales



Computational Cognitive Science

Bayesian perception

Diffusion models

Computational Cognitive Neuroscience

Connectivity analysis

Neural networks

Computational Neuroscience

Neural coding

Neuronal models

Taxonomy based on purposes

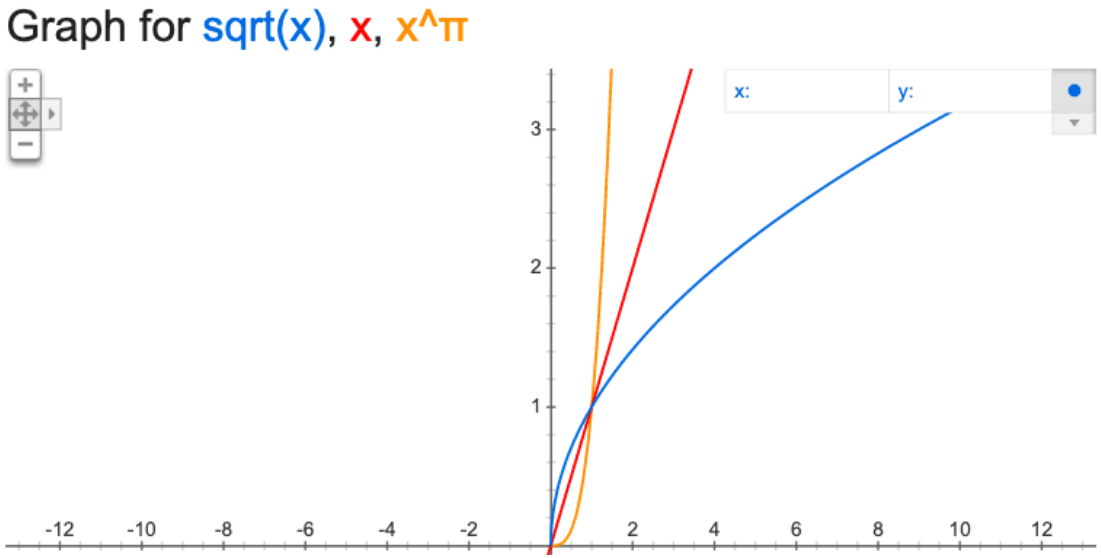
Conceptual Models

Making sense of data qualitatively; guiding new experiments

Mathematical Models

A compact and precise summary of data

- Descriptive (What)
- Interpretive (Why)
- Mechanistic (How)



Computer Simulations

Allowing concrete visualization of abstract mathematics

Correspondence Principle : A Hope

Synaptic redistribution/Synaptic scaling/STDP

Molecules/Ion-Channels → Neurons → Networks → Behavior

Mechanistic
↓
Descriptive → Mechanistic
↓
Descriptive → Mechanistic
↓
Descriptive → Mechanistic

Utility of Models: Data Reduction

=

17 Equations That Change the World

1.	Pythagora's Theorem	$a^2 + b^2 = c^2$	Pythagoras, 530 BC
2.	Logarithms	$\log xy = \log x + \log y$	John Napier, 1610
3.	Calculus	$\frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$	Newton, 1668
4.	Law of Gravity	$F = G\frac{m_1 m_2}{d^2}$	Newton, 1687
5.	The Square Root of Minus One	$i^2 = -1$	Euler, 1750
6.	Euler's Formula for Polyhedra	$F - E + V = 2$	Euler, 1750
7.	Normal Distribution	$\Phi(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	C.F. Gauss, 1810
8.	Wave Equation	$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$	J. d'Almbert, 1746
9.	Fourier Transform	$\hat{f}(\in) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i x \in} dx$	J.Fourier, 1822
10.	Navier-Stokes Equations	$\rho\left(\frac{\partial v}{\partial t} + v \cdot \nabla v\right) = -\nabla p \quad \nabla \cdot T + f$	C. Navier, G. Stokes, 1845
11.	Maxwell's Equations	$\nabla E = \frac{\rho}{\in_0} \quad \nabla \times E = -\frac{1}{c}\frac{\partial H}{\partial t}$ $\nabla H = 0 \quad \nabla \times H = \frac{1}{c}\frac{\partial E}{\partial t}$	J.C Maxwell, 1865
12.	Second Law of Thermodynamics	$dS \geqslant 0$	L. Boltzmann, 1874
13.	Relativity	$E = mc^2$	Einstein, 1905
14.	Schrodinger's Equation	$i\hbar\frac{\partial}{\partial t}\Psi = \hat{H}\Psi$	E. Schrodinger, 1927
15.	Information Theory	$H = -\sum p(x) \log p(x)$	C. Shannon, 1949
16.	Chaos Theory	$x_{t+1} = kx_t(1 - x_t)$	Robert May, 1975
17.	Black-Scholes Equation	$\frac{1}{2}\sigma S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + \frac{\partial V}{\partial t} - rV = 0$	F. Black, M. Scholes, 1990

All models are wrong (1/3)

But some are useful (Box & Draper, 1987)

The point of modeling is to characterize defining features

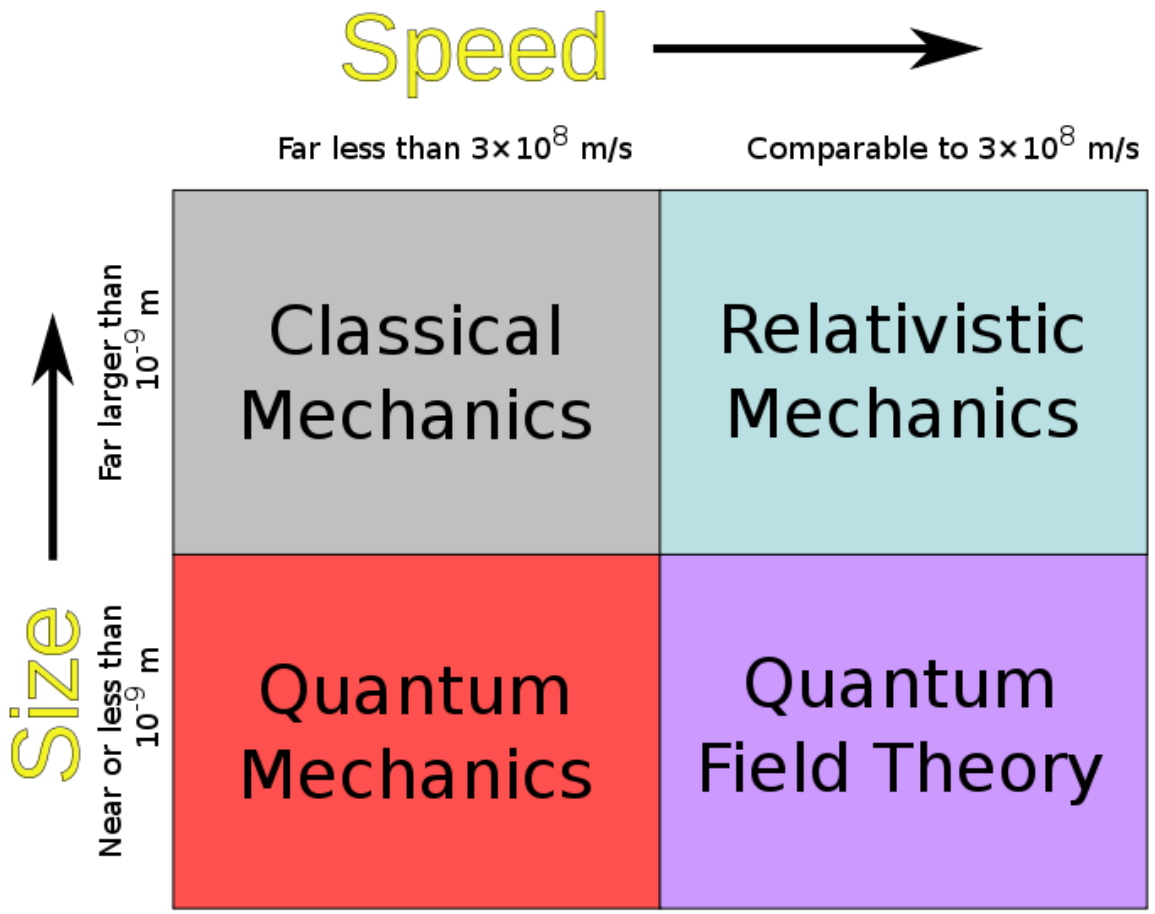
All models are wrong (2/3)

The most realistic model is not necessarily the most useful one

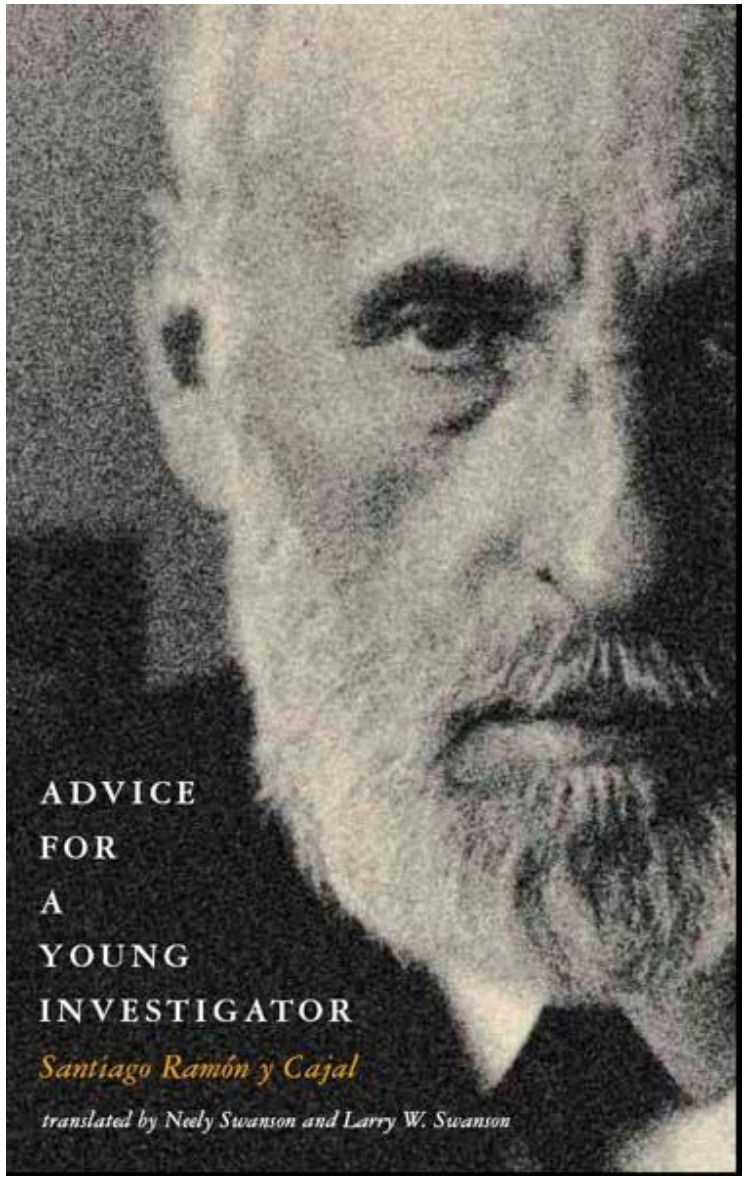
Irrelevant details mask the key components and their interactions

All models are wrong (3/3)

Models/theories are just interpretations of (data=truths)



Advice for the Young



"A scholar's positive contribution is measured by the sum of the original data that he contributes. Hypotheses come and go but data remain. Theories desert us, while data defend us."

"The beginner should devote maximal effort to discovering original facts by making precise observations, carrying out useful experiments, and providing accurate descriptions."

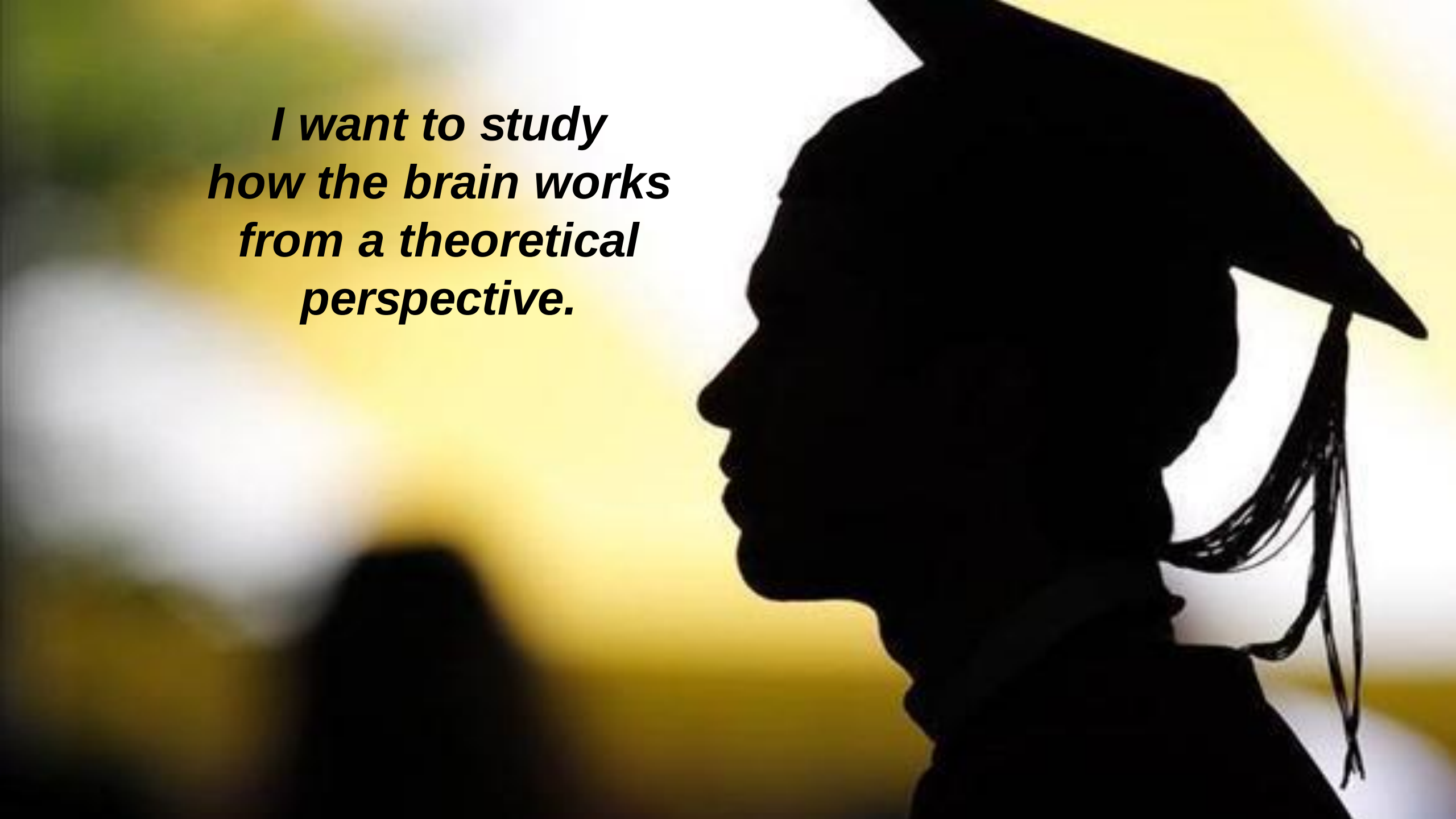
Statistics on Nobel Prizes

Categorization of Nobel Science Awards, 1991–2011

	Medicine	Chemistry	Physics	Total
Theory	2	4	8	14
Method	21	22	20	63

Theory will always get blown away by data.

- Jack Gallant


I want to study how the brain works from a theoretical perspective.