



# Future of Probability

## Chris Piech + Jerry Cain



# 2021 Abel Prize (~Nobel Prize Math) **Two Computer Scientists**

One winner Avi Wigderson

Major theme: randomness in computer science

## Eg Zero knowledge proof

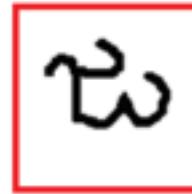
<https://www.nytimes.com/2021/03/17/science/abel-prize-mathematics.html>

Abel Prize Chair: “This prize is on the applied side, toward computer science,” Dr. Munthe-Kaas said. “But it’s deep mathematics.”

# Open Problem: One Shot Learning

B Lake, R Salakhutdinov, J Tenenbaum. Science 2015.

Human-level concept learning through probabilistic program induction.



କୁ	ବୁ	ଗୁ	ଚୁ	ମୁ
ଫୁ	ପୁ	ଛୁ	ତୁ	ଦୁ

Current deep learning methods are not enough to move the needle as far as we want, **especially on socially relevant problems** that often do not have the benefit of massive public datasets. The best new ideas are coming from probability theory



# Open Problem: One Shot Learning

B Lake, R Salakhutdinov, J Tenenbaum. Science 2015.

Human-level concept learning through probabilistic program induction.



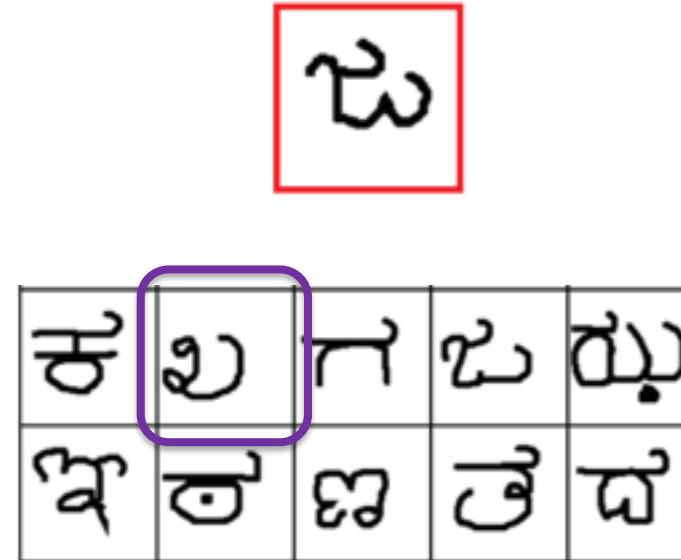
Current deep learning methods are not enough to move the needle as far as we want, **especially on socially relevant problems** that often do not have the benefit of massive public datasets. The best new ideas are coming from probability theory



# Open Problem: One Shot Learning

B Lake, R Salakhutdinov, J Tenenbaum. Science 2015.

Human-level concept learning through probabilistic program induction.



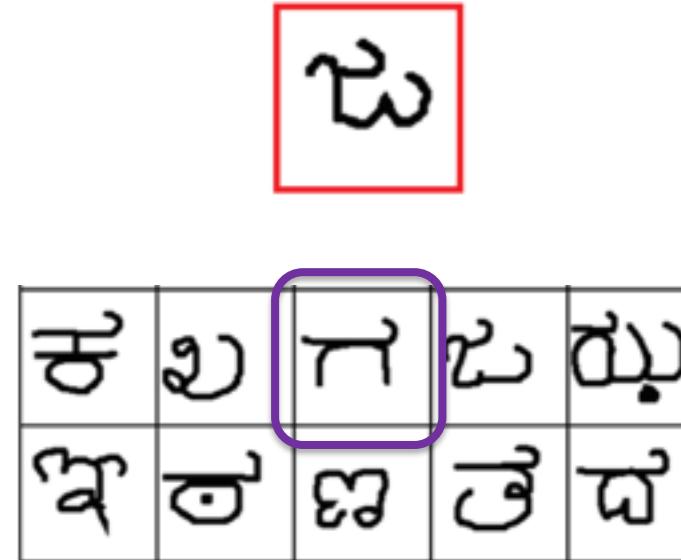
Current deep learning methods are not enough to move the needle as far as we want, **especially on socially relevant problems** that often do not have the benefit of massive public datasets. The best new ideas are coming from probability theory



# Open Problem: One Shot Learning

B Lake, R Salakhutdinov, J Tenenbaum. Science 2015.

Human-level concept learning through probabilistic program induction.



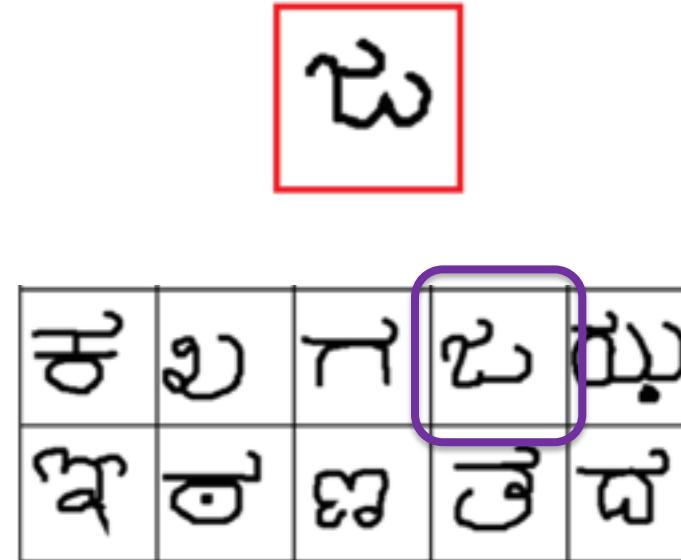
Current deep learning methods are not enough to move the needle as far as we want, **especially on socially relevant problems** that often do not have the benefit of massive public datasets. The best new ideas are coming from probability theory



# Open Problem: One Shot Learning

B Lake, R Salakhutdinov, J Tenenbaum. Science 2015.

Human-level concept learning through probabilistic program induction.



Current deep learning methods are not enough to move the needle as far as we want, **especially on socially relevant problems** that often do not have the benefit of massive public datasets. The best new ideas are coming from probability theory



Today  
**Probabilities digital future**

Let me tell you a story about one particular problem.

It will give you a sense of what it takes to do research in **applied probability**, and our desire to solve the problems, will lead us to deep **theoretical** challenges in modern AI.

**Application -> Theory**

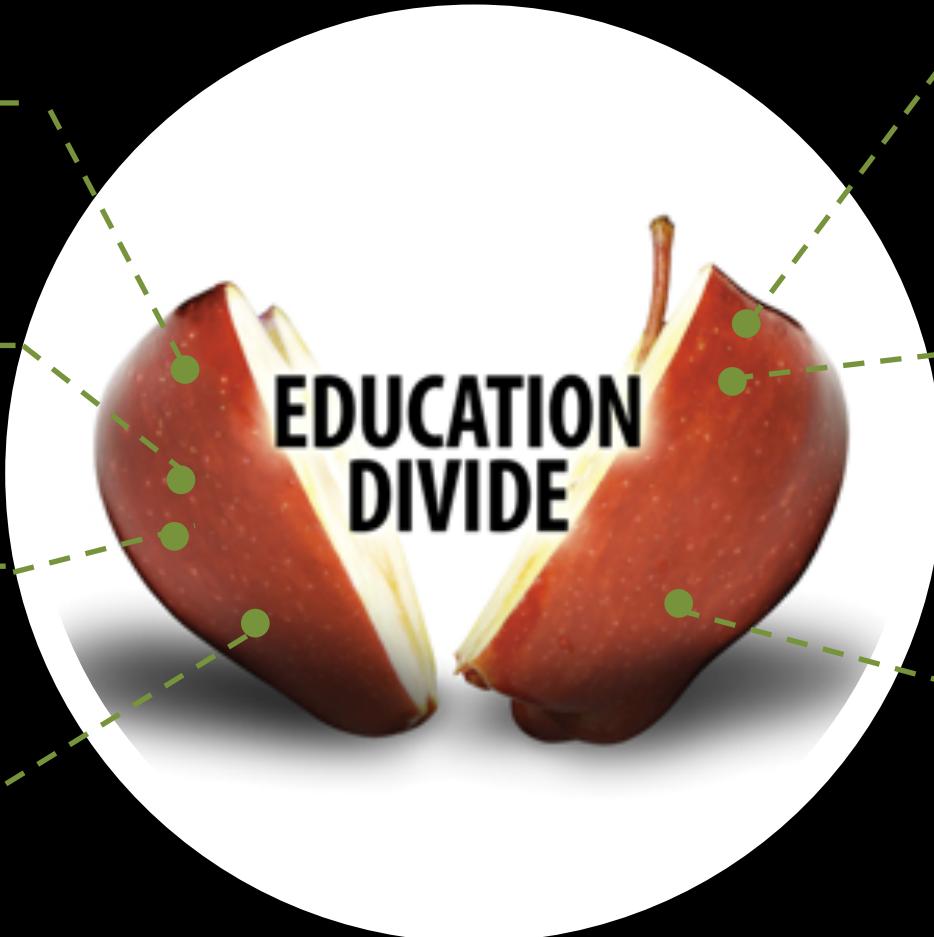
# Quality Education Gap

90% of children  
enroll in primary  
education [1]

40% in secondary  
education [1]

20% in tertiary  
education [1]

Dramatic quality  
differences



375 million workers need to  
be retrained by 2030 [3]

Half a million unfilled  
computer science jobs  
(60% of STEM jobs) [2]

For all learners we  
want *quality*.

[1] *World development indicators 2015*. World Bank Publications, 2015.

[2] USA Bureau of Labor Statistics Employment Projections , 2016.

[3] Jobs lost, jobs gained. McKinsey Global Institute, 2017.



# Smart Phone Access

Advanced Economies



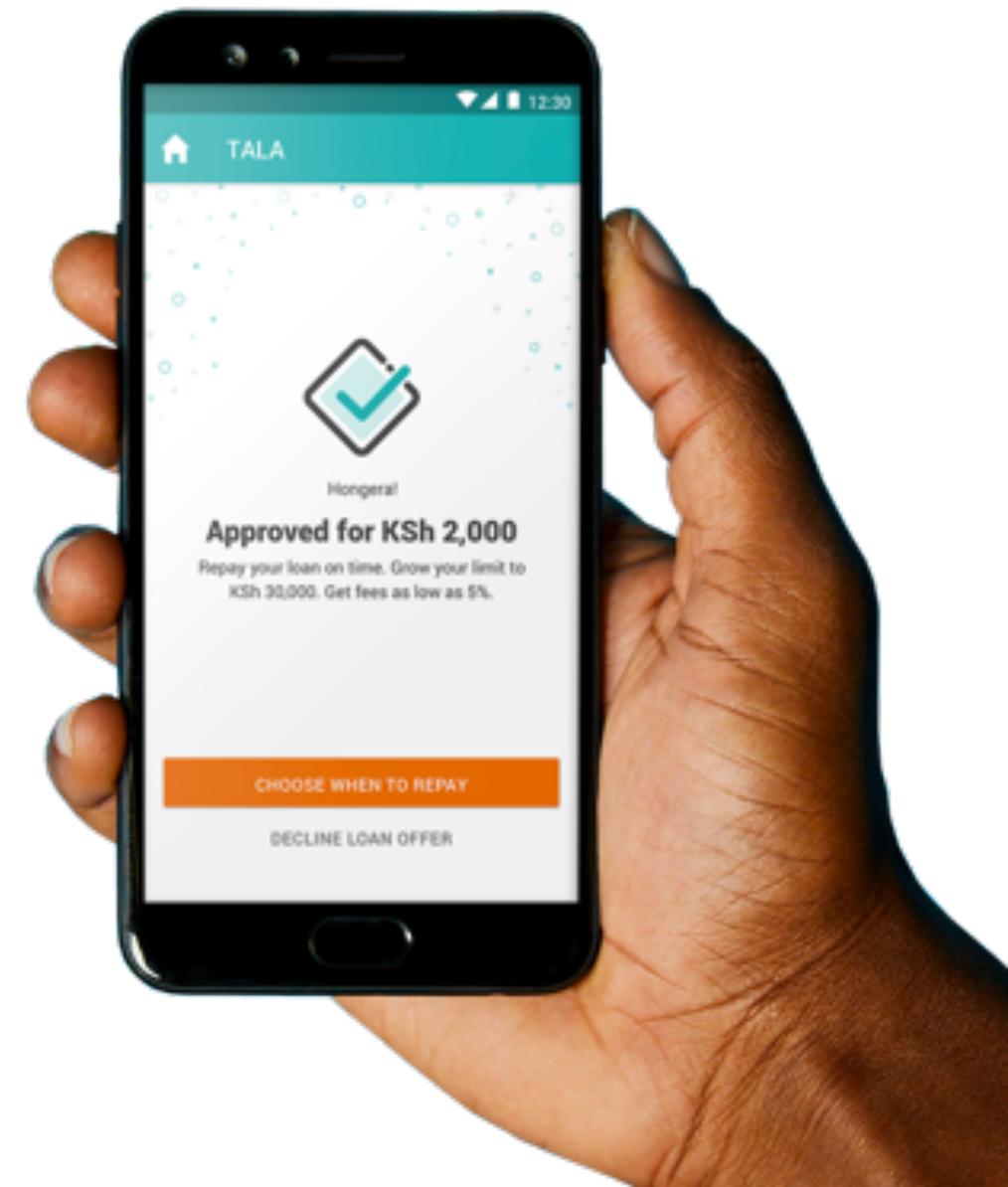
Emerging Economies



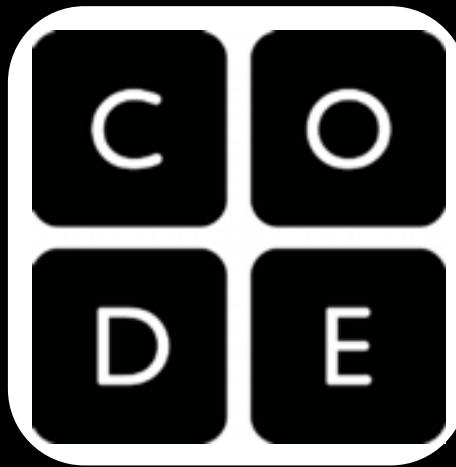
Smartphone

Mobile

No phone



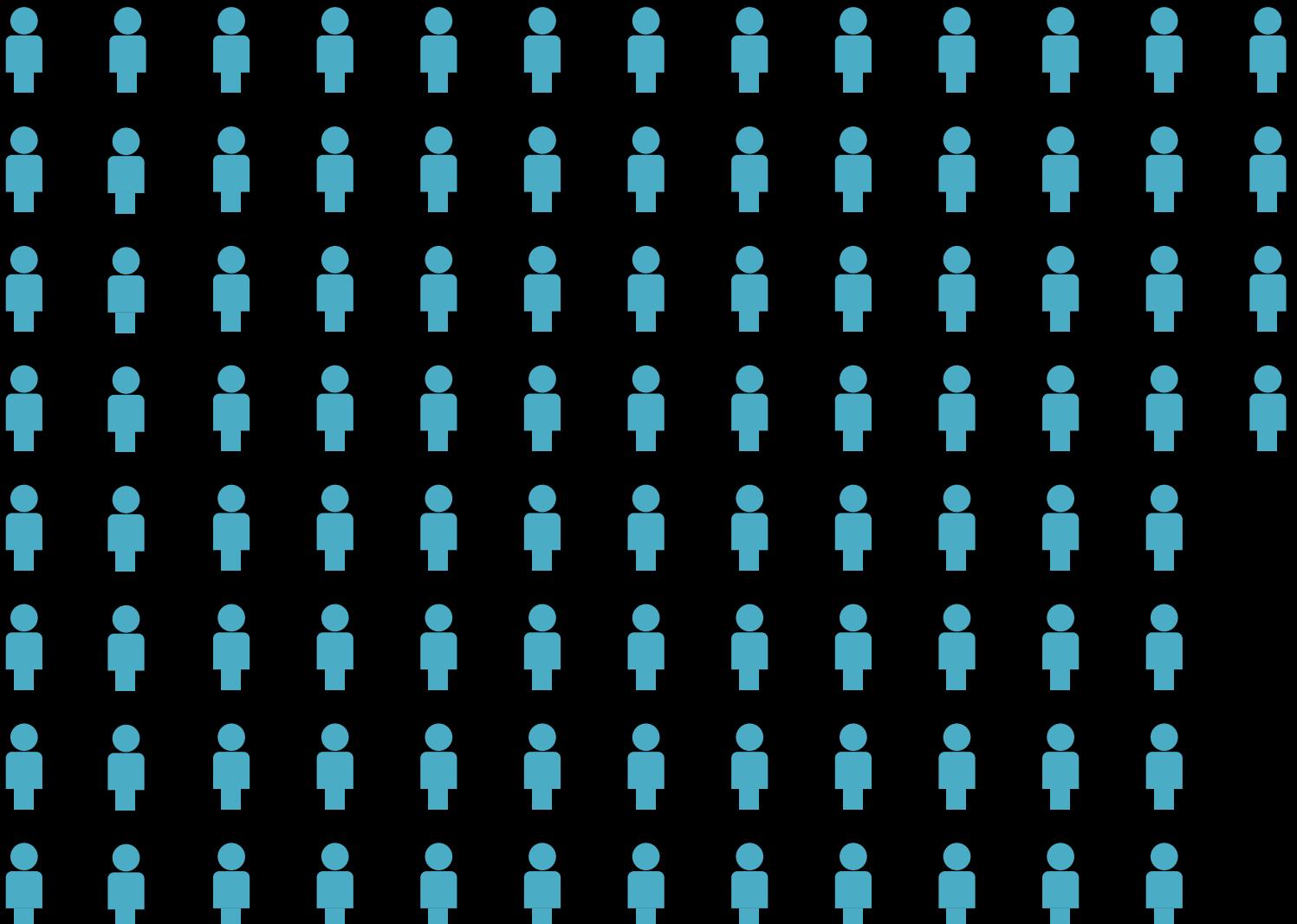
# Unprecedented Data



Over 50 million learners



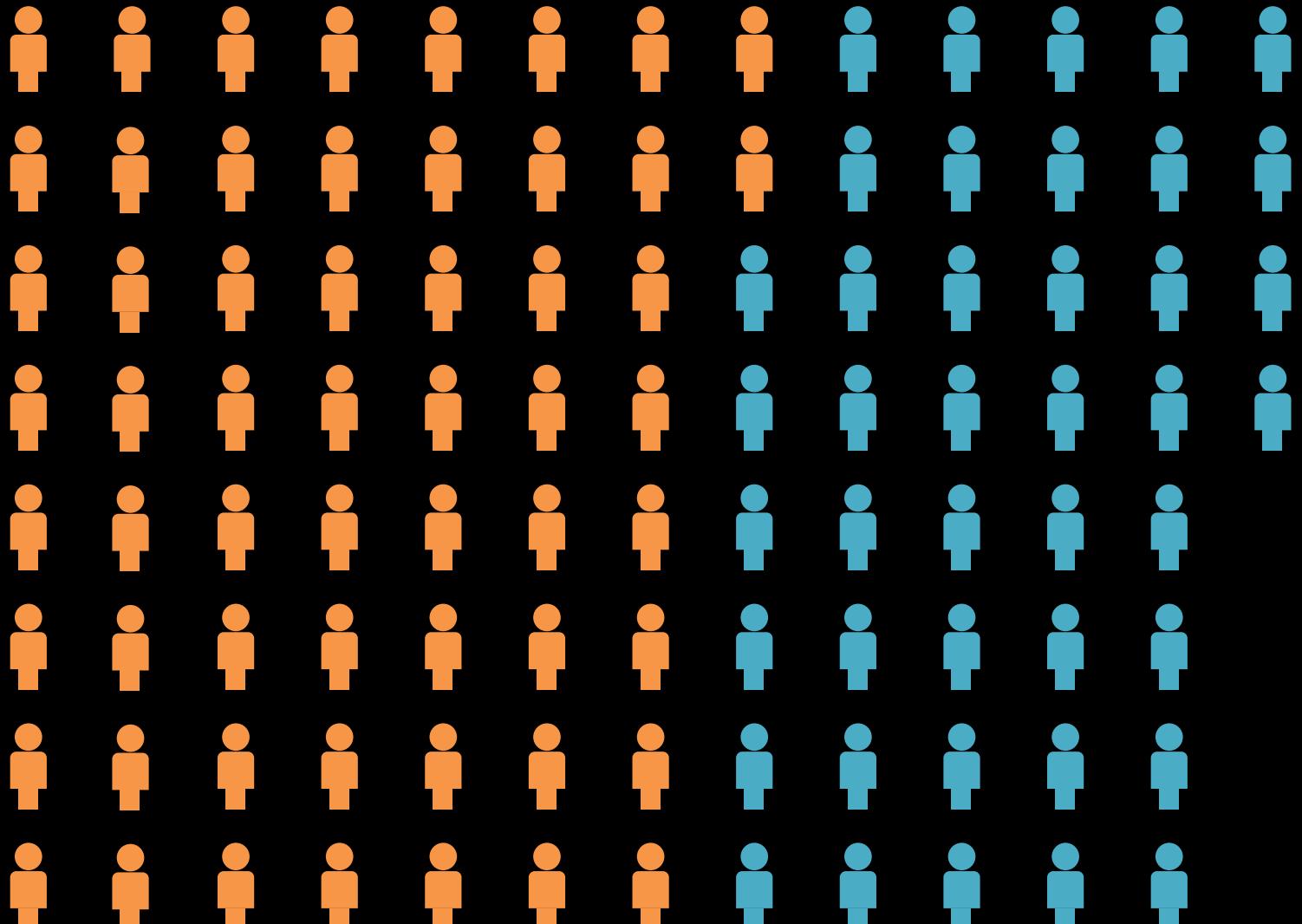
# US K-12 Students



= 500,000 learners



# Code.org Students



= 500,000 learners



# Code.org by the numbers

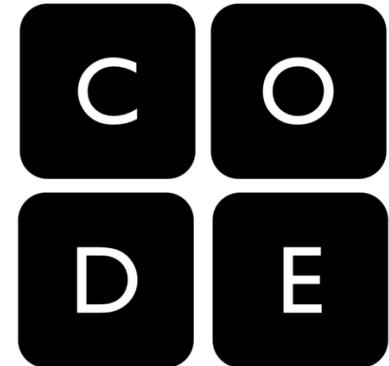
**1,234,127** teachers

**42M** unique enrolled students

Used in **180+** countries

**832M** hour of code sessions

**4** papers publish with our lab

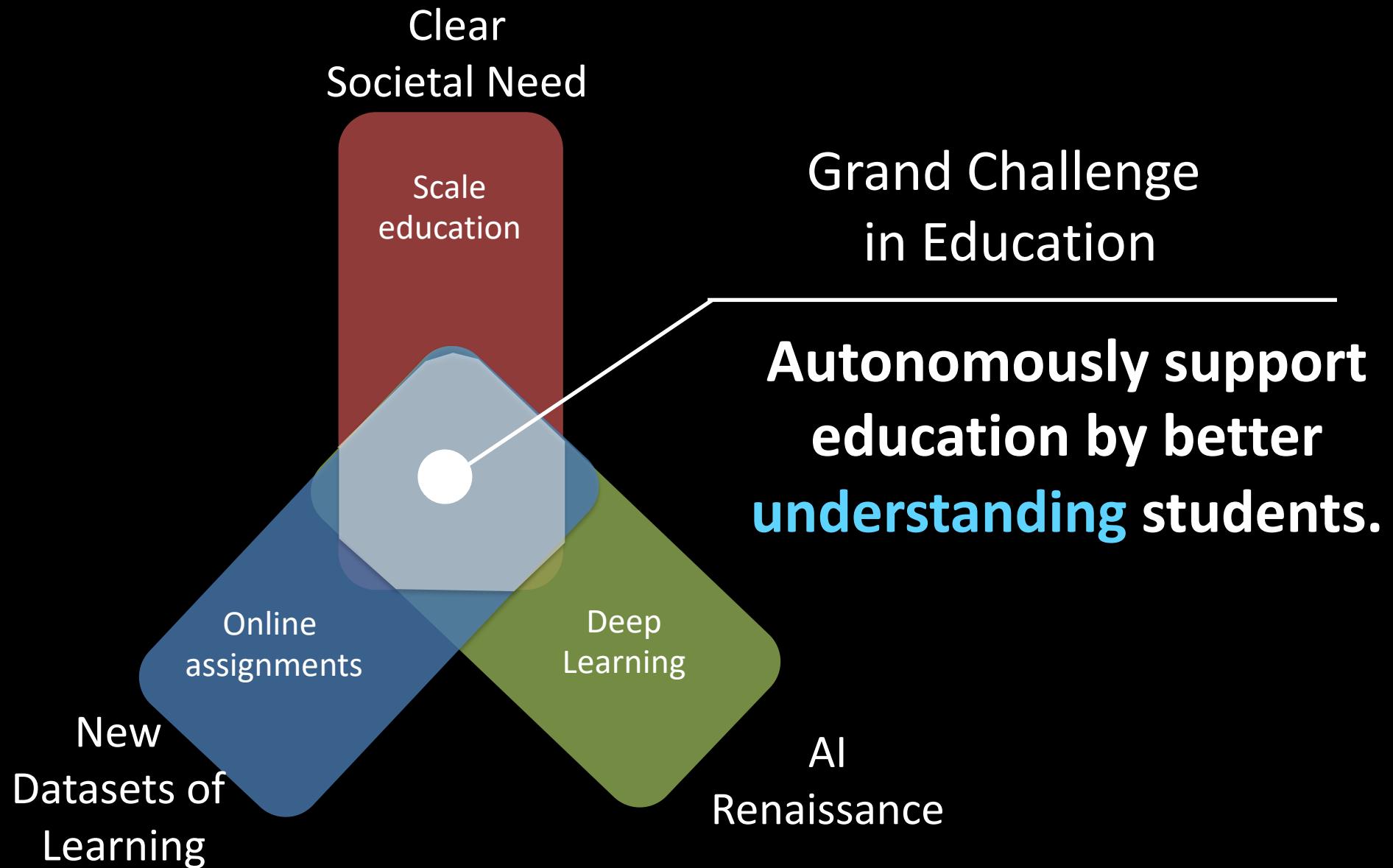


50M K12 students in the US



# Speech Recognition





# Feedback is Labor Intensive



Online classes have not solved the feedback problem [1].



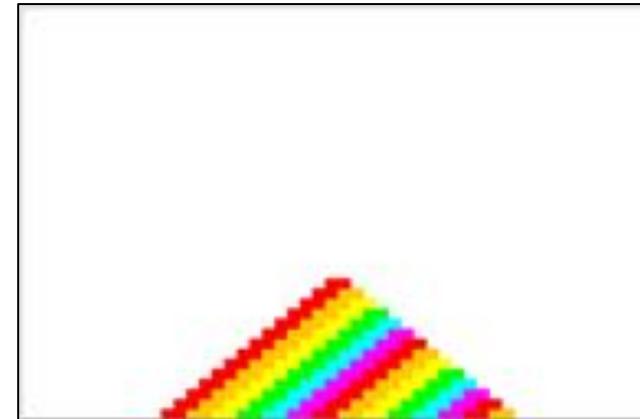
[1] Deconstructing Disengagement. Analysing learner subpopulations in MOOCs. Kizilcec, Piech, Schneider. Over 600 citations since 2013



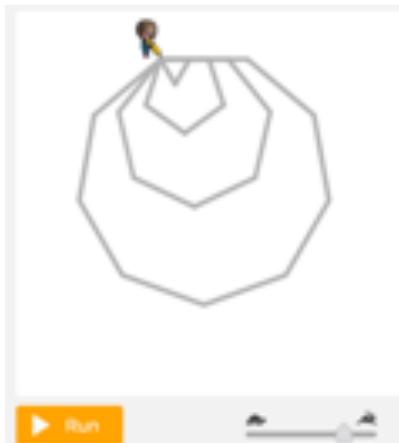
# Many domains of student work

$$\begin{aligned} 9 \times 6 &= (10 - \boxed{\phantom{0}}) \times 6 \\ &= 10 \times 6 - \boxed{\phantom{0}} \times 6 \\ &= 60 - \boxed{\phantom{0}} \\ &= \boxed{\phantom{0}} \end{aligned}$$

```
when run  
repeat until [acorn]  
  do [if path ahead  
    do [turn left  
    else move forward]]
```



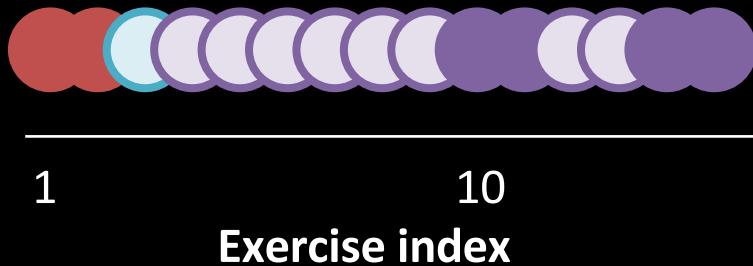
Why did the original pilgrims come to America?



# **Chapter 0: Always start simple**

# First deep learning for education

KHAN  
Student

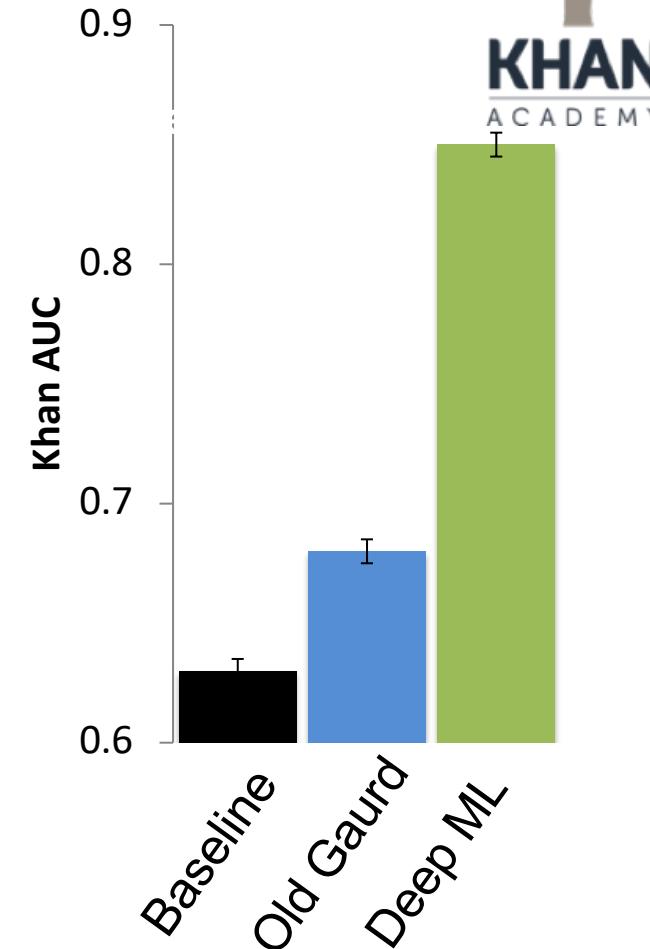


## Exercise Type:

- Solving for x-intercept
- Solving for y-intercept
- Graphing linear equations
- Square roots
- Slope of a line

## Answer:

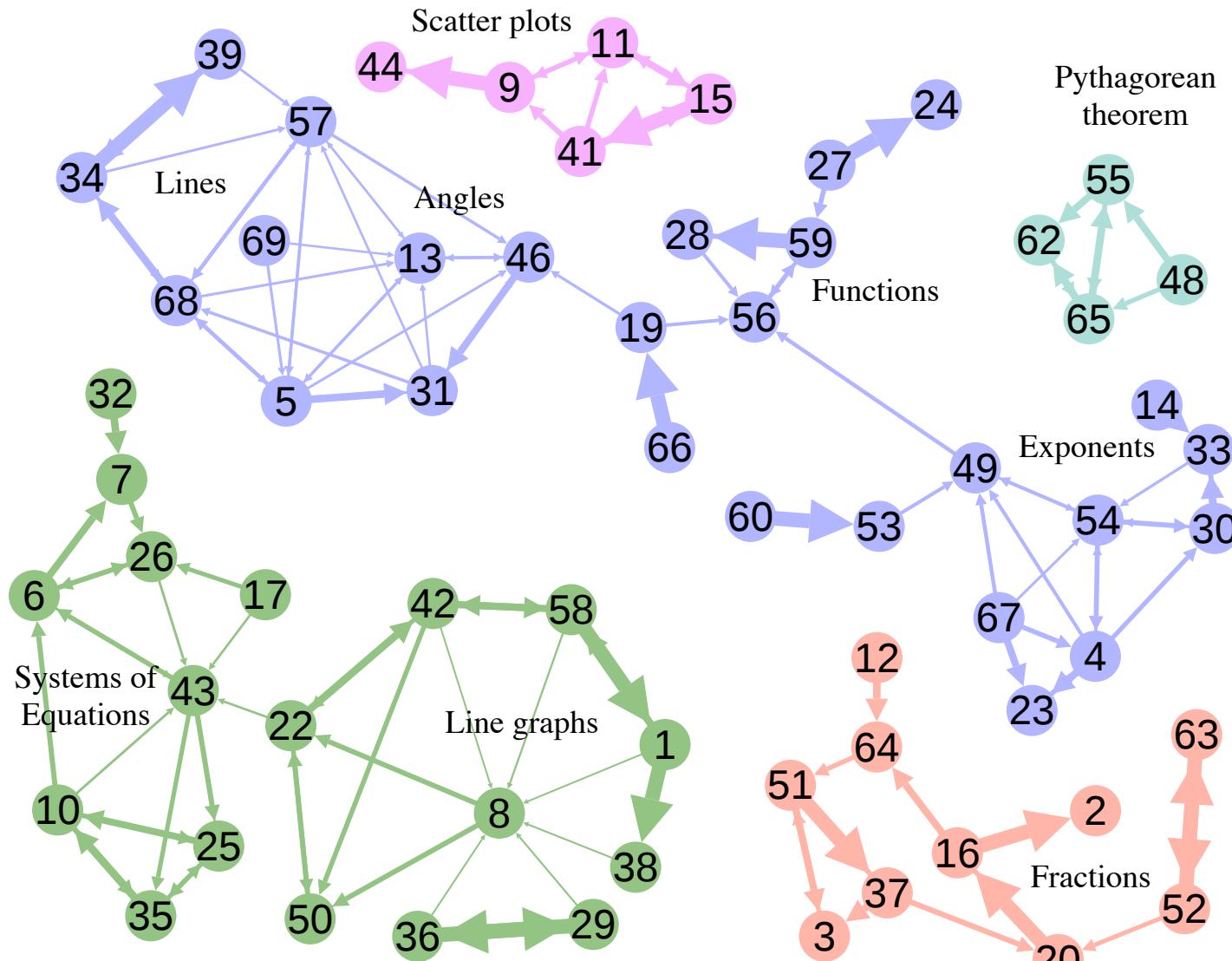
- Correct
- Incorrect



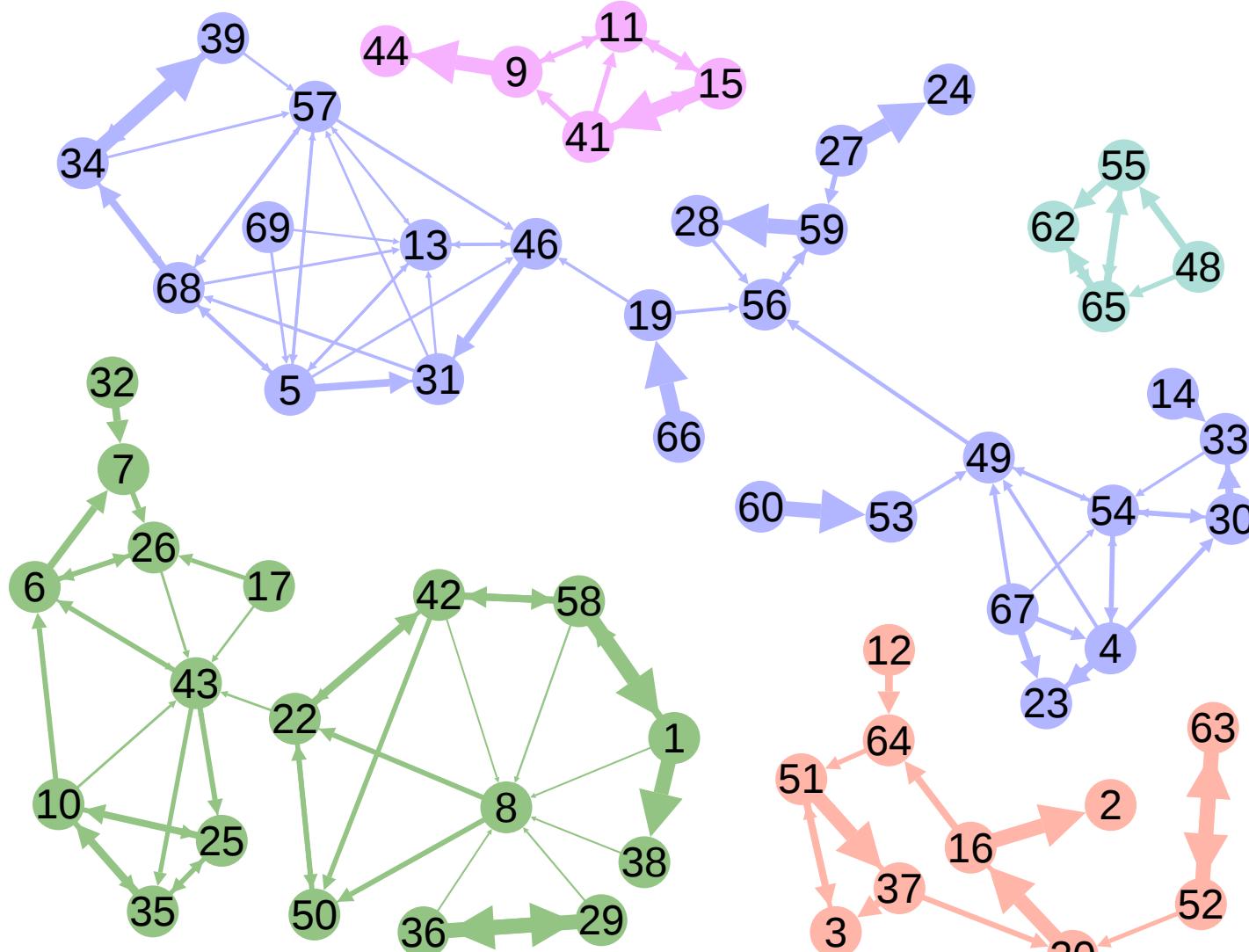
# Old Problem



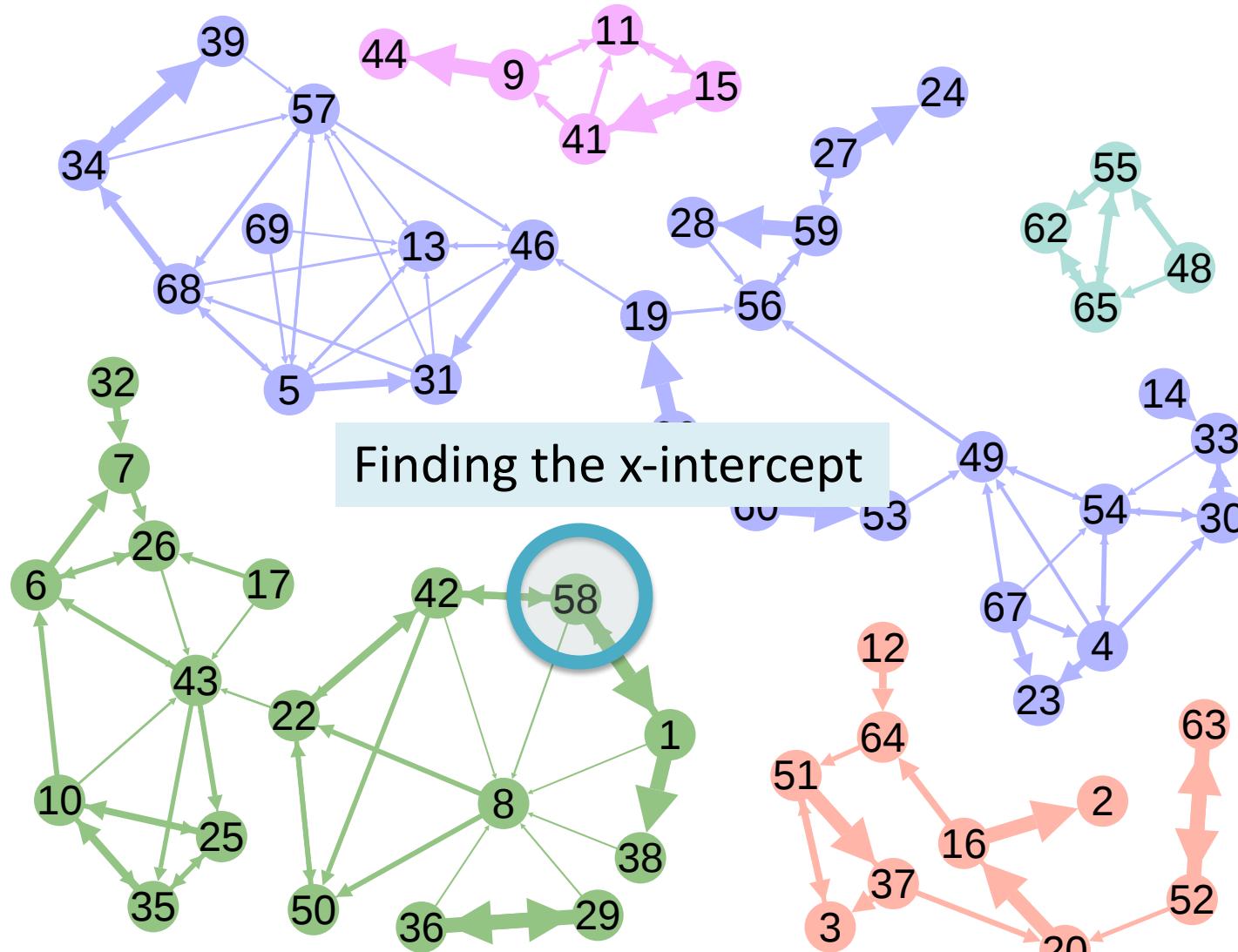
# Learns Concept Relationships



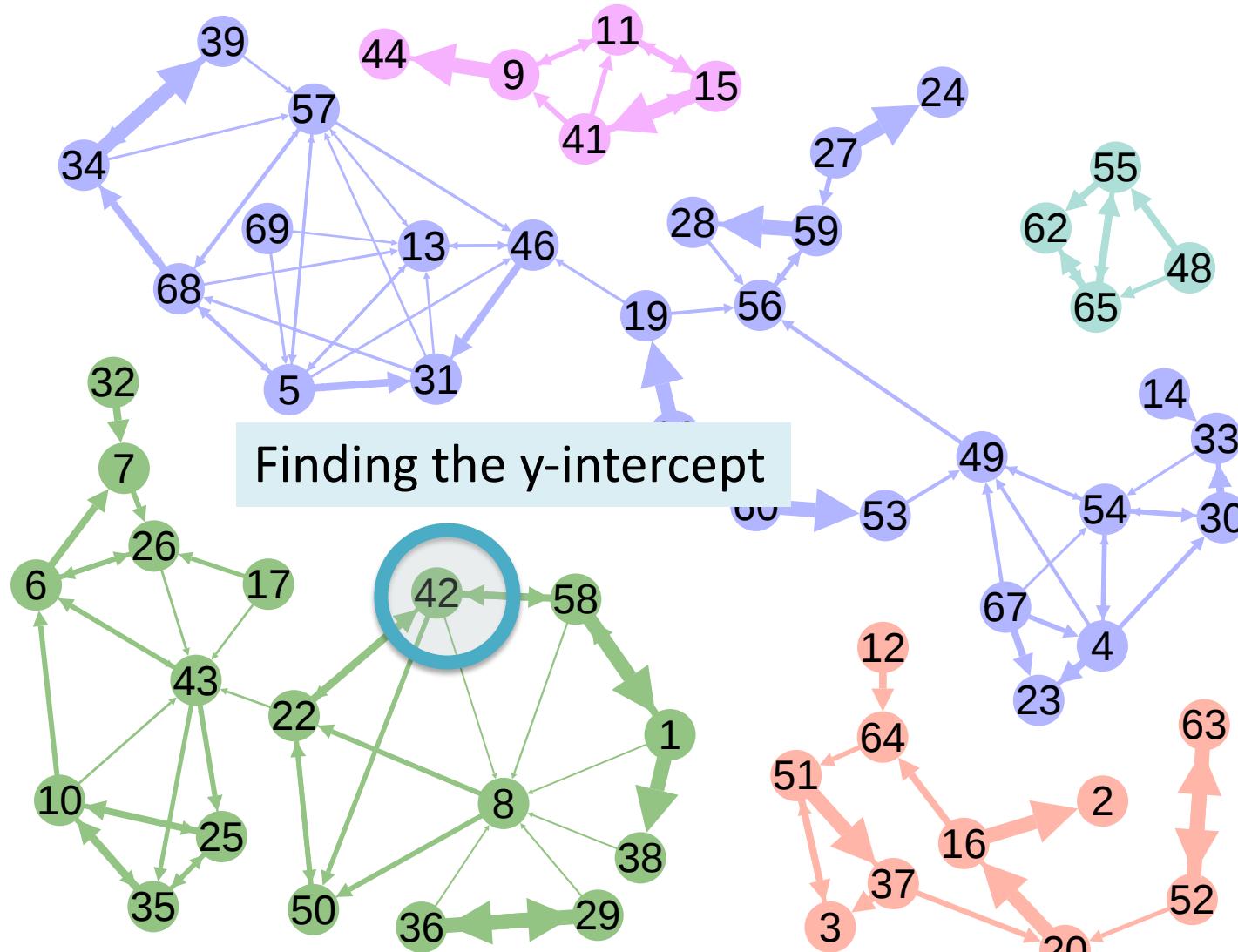
# Learns Concept Relationships



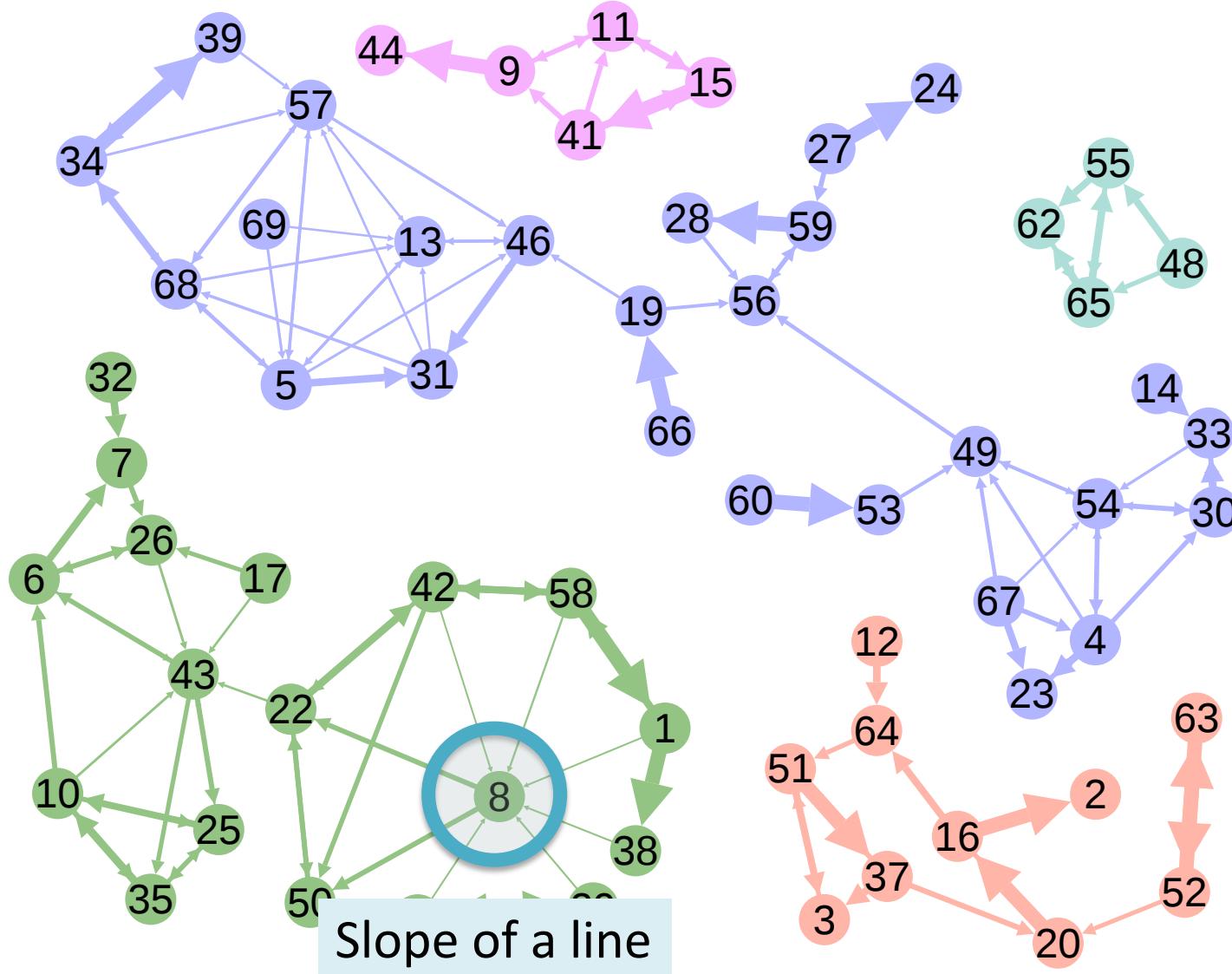
# Learns Concept Relationships



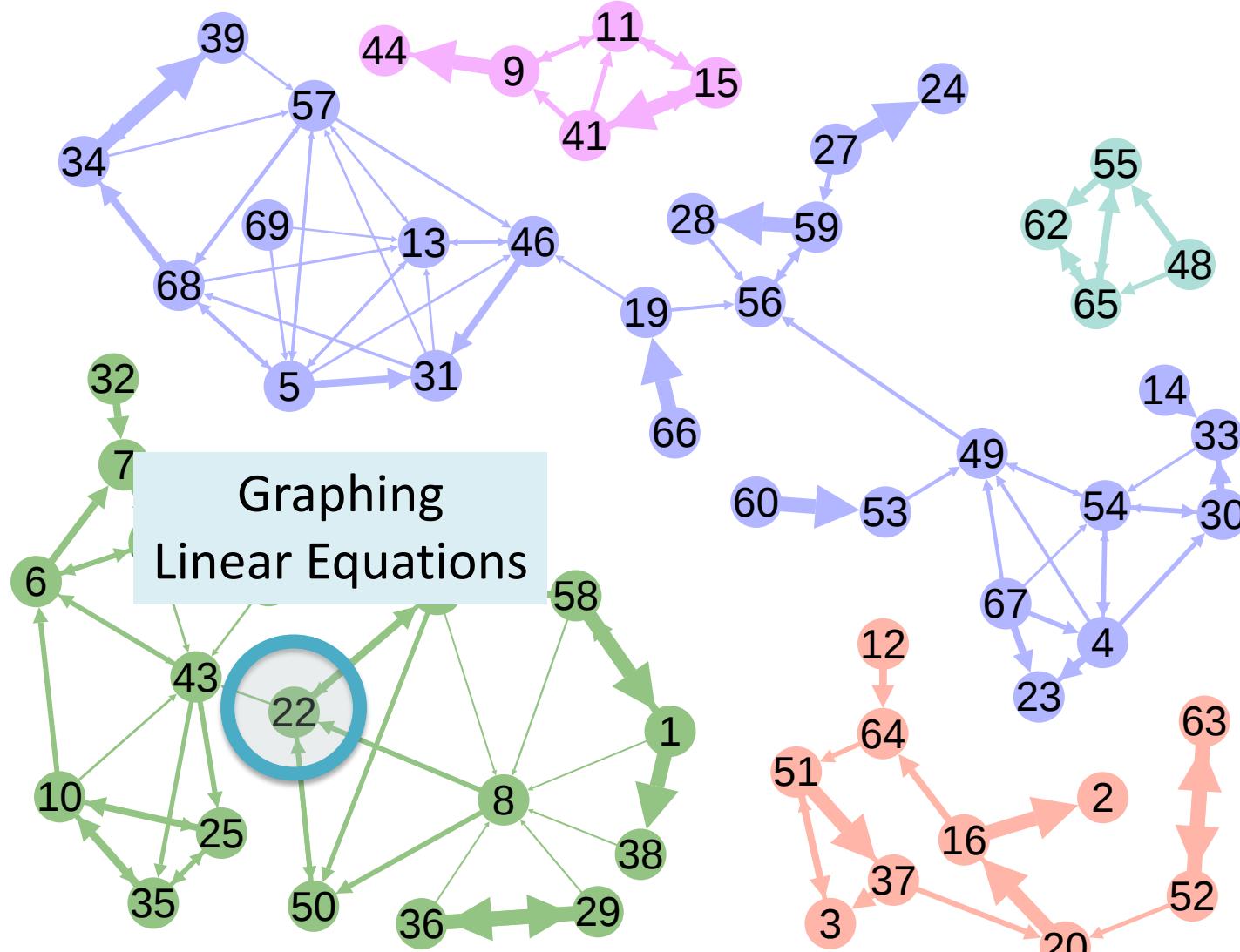
# Learns Concept Relationships



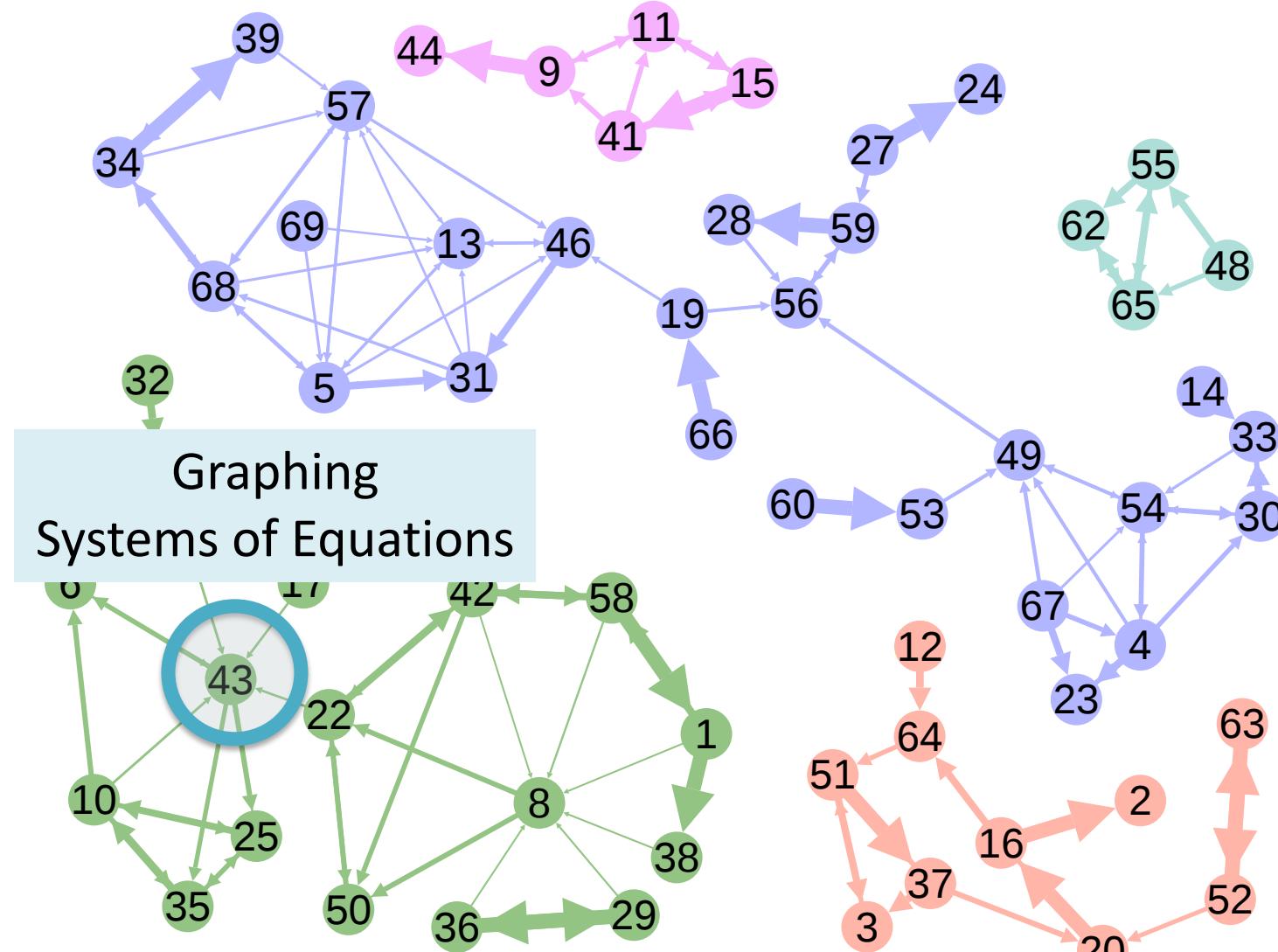
# Learns Concept Relationships



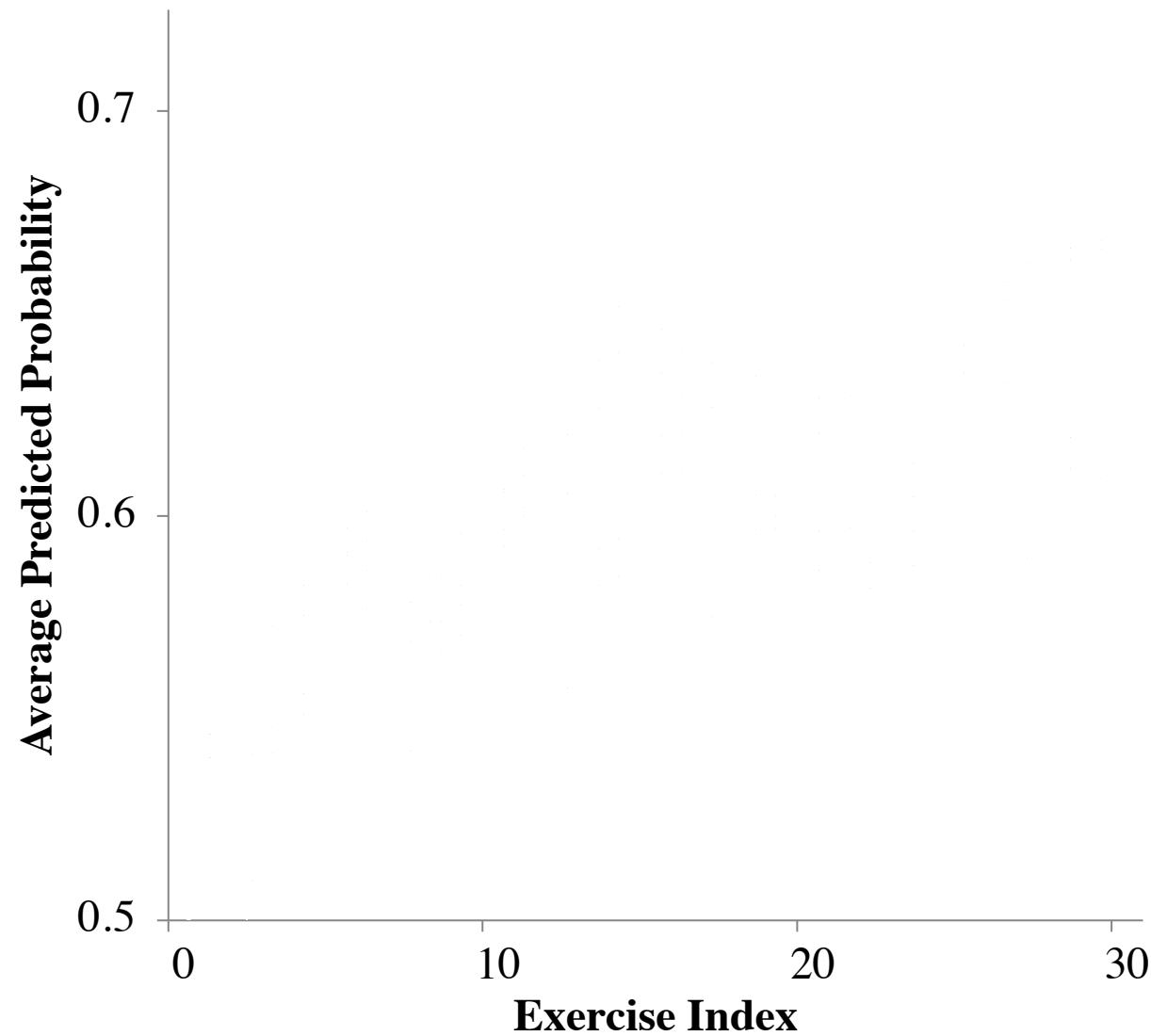
# Learns Concept Relationships



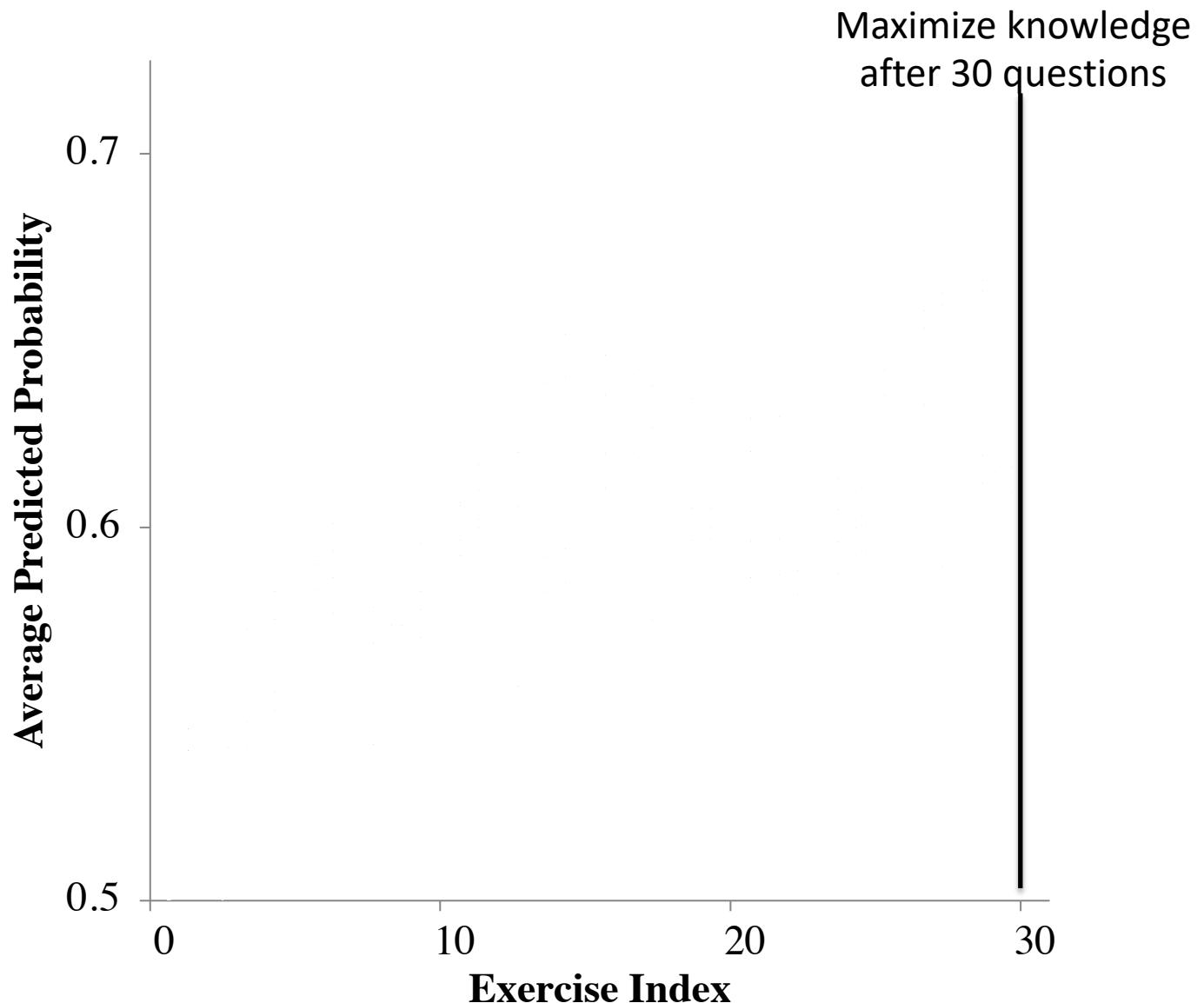
# Learns Concept Relationships



# Optimal Teaching

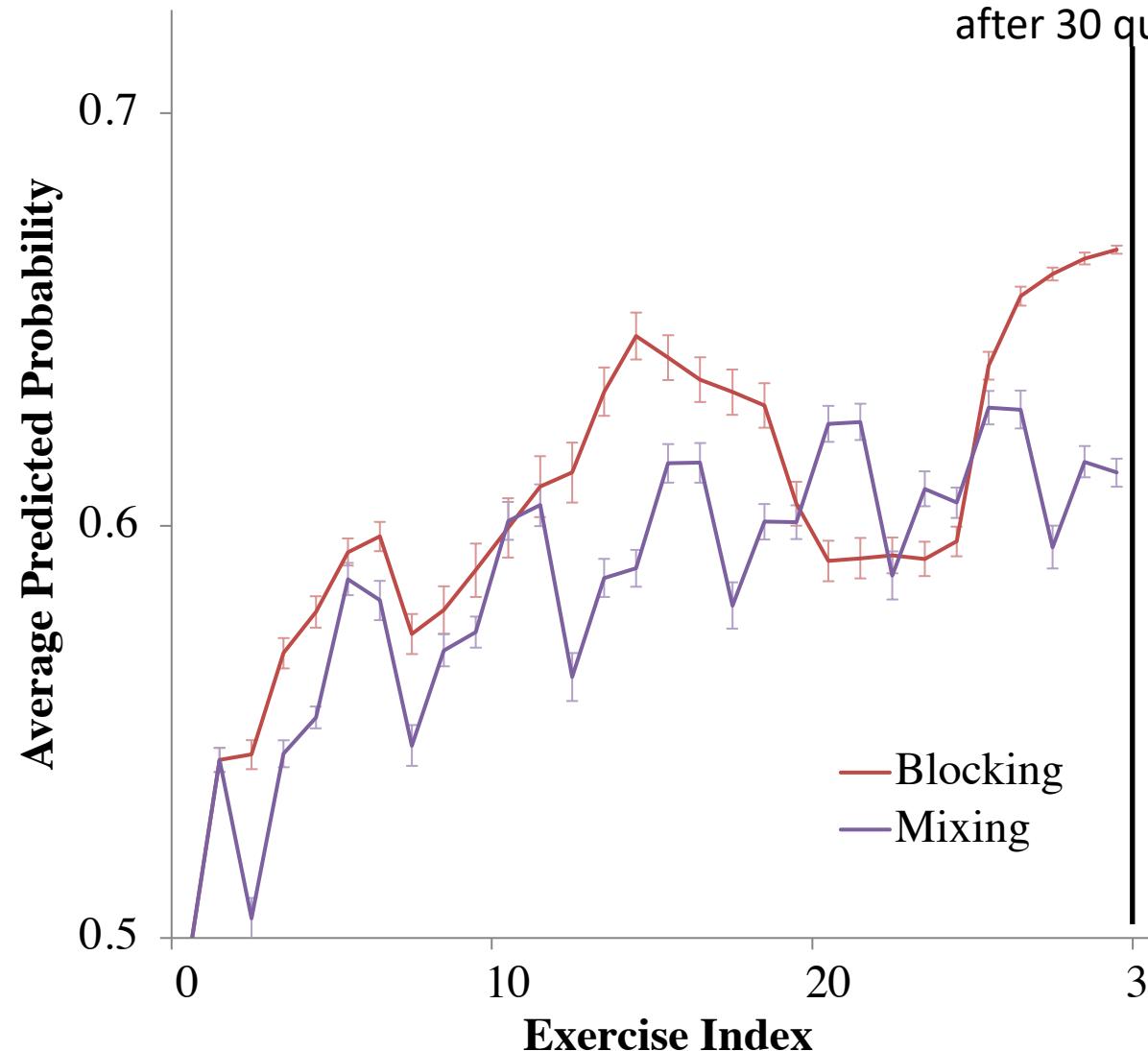


# Optimal Teaching



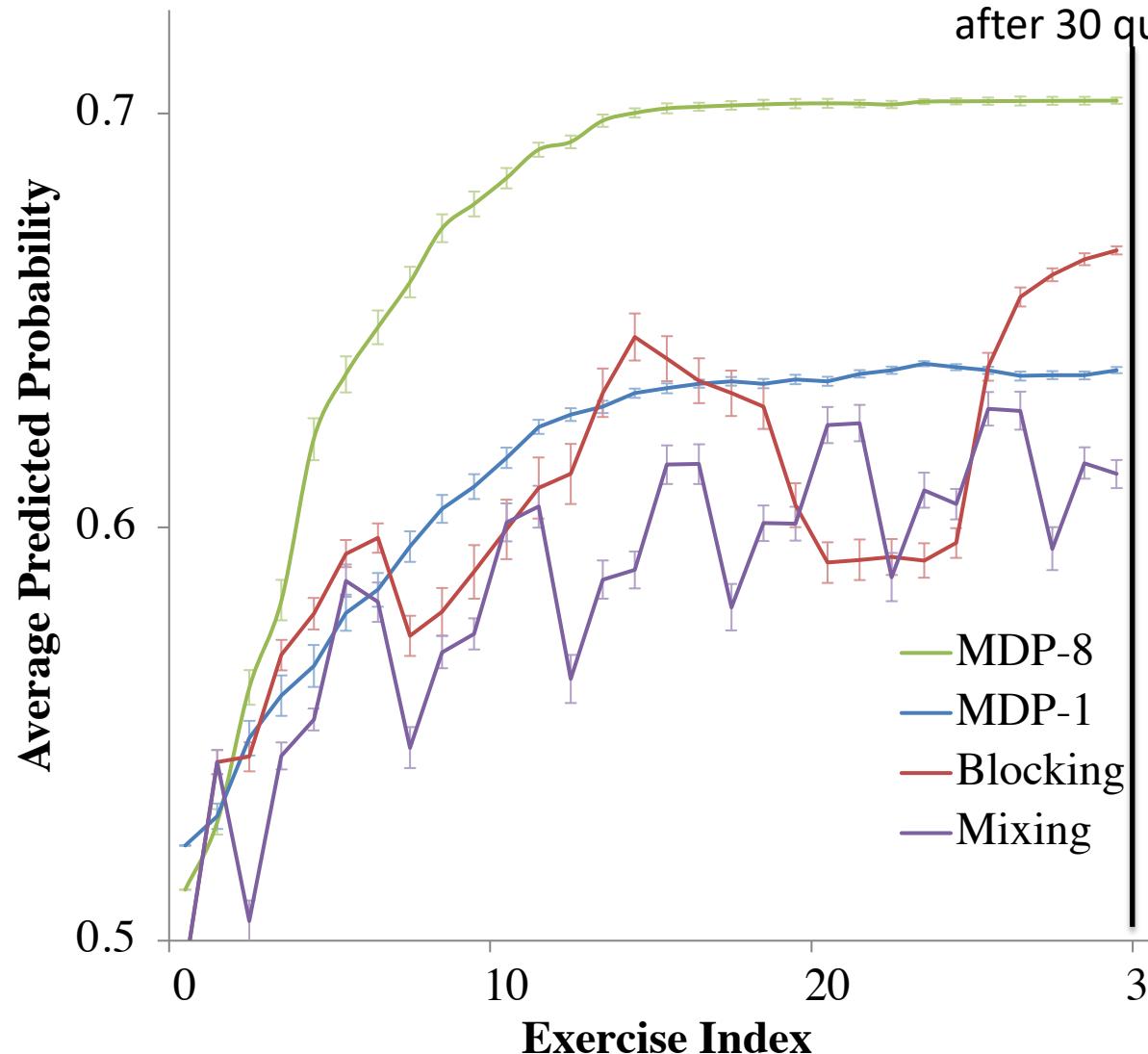
# Optimal Teaching

Maximize knowledge  
after 30 questions



# Optimal Teaching

Maximize knowledge  
after 30 questions



We truly would rather move beyond  
correct / incorrect



# Some domains are very hard

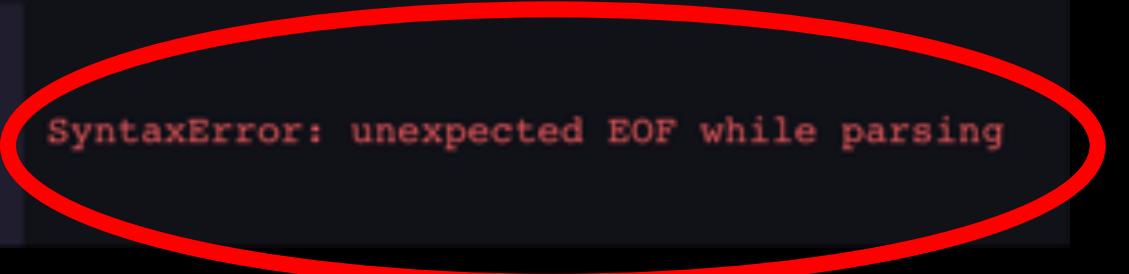
Introduction to Python      Upgrade to Pro      ⚡      📲

script.py

```
1 my_name = "Codecademy"
2 print("Hello and welcome " + my_name + "!")
3
```

File "script.py", line 3

SyntaxError: unexpected EOF while parsing



# Can you understand this code?

Top Secret    Not Secure

Question Solution

## Instructions

- If there are many moves, focus on the first one
- Random code strategy is for when the student seems to be trying things randomly
- Lookout for students who don't get nesting or pre/post conditions. Often extra blocks in a body is an indication that they don't get that the post of the loop has to match the precondition

## Question



**import** code.org.\*;

```
public class MySoln {
    public void run() {
        move(50);
        for(int i=0; i<4; i++){
            if(frontIsClear()) {
                turnLeft(90);
            }
            for(int j=0; j<i; i++){
                move(i * 20);
                turnRight(120);
                move(10);
            }
        }
    }
}
```

Student 0

## Label Console

✓ Num Done: 8273

**Strategy**

Beeper Boundary (most people do this)  
 Triangle Strategy  
 Recursive Strategy

**Looping**

Correct use of looping  
 Doesn't use a while  
 Doesn't have correct stop condition

**Stanford TAs label**  
**800 submissions**

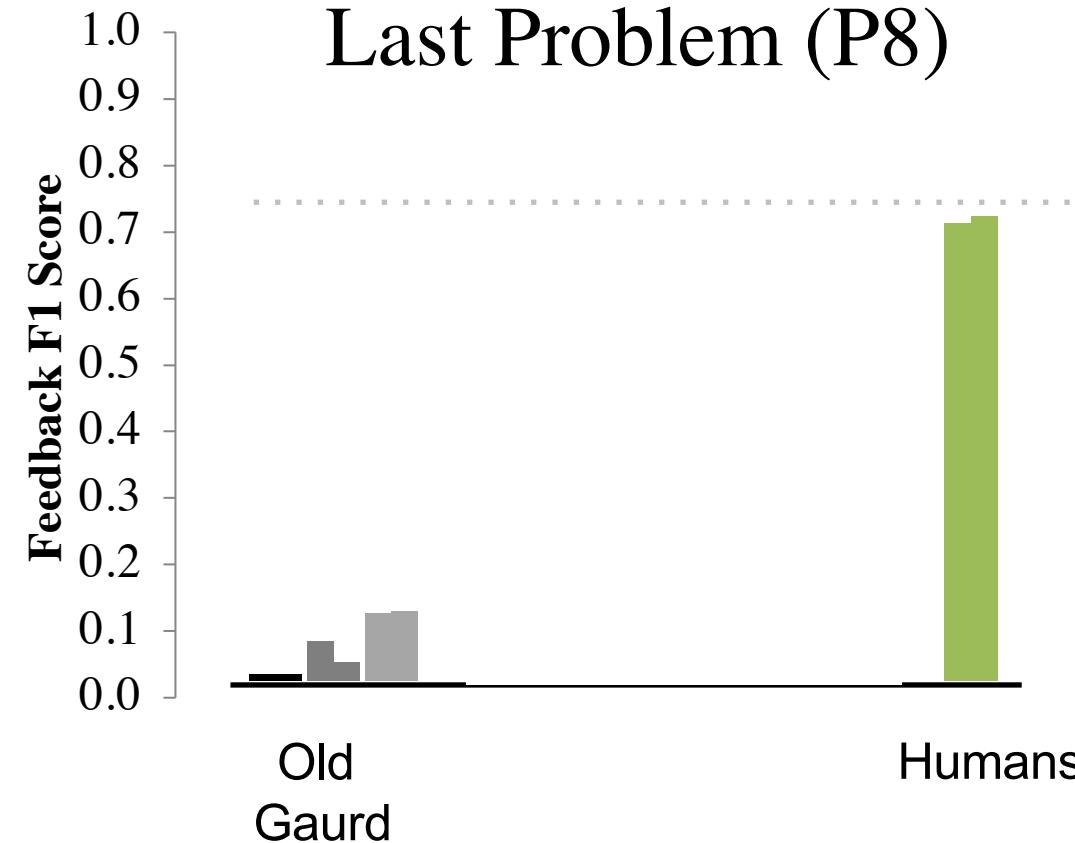
Loop post condition doesn't match precondition  
 Repetition of bodies

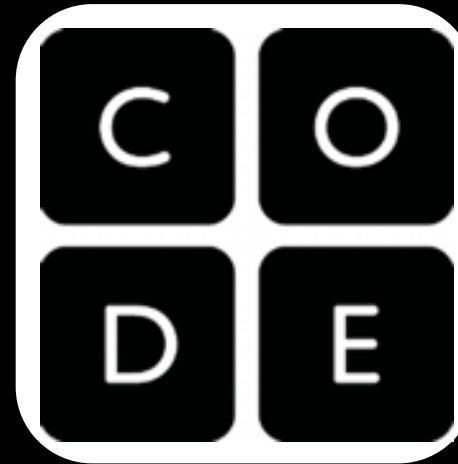
**Cleanup**

Record label

# Traditional Deep Learning Doesn't Work

Label student code





Can we provide feedback  
by dynamic analysis?



...

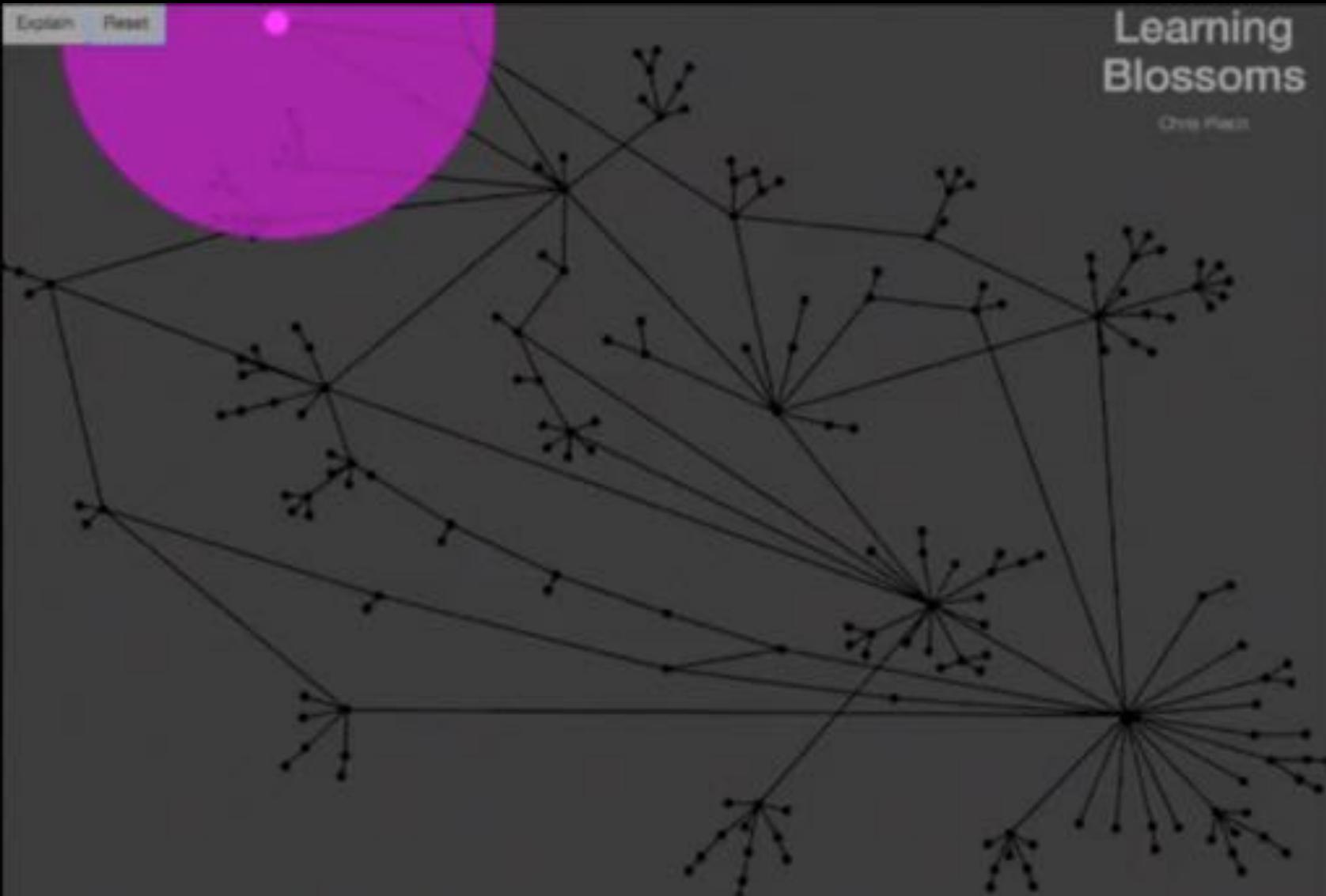


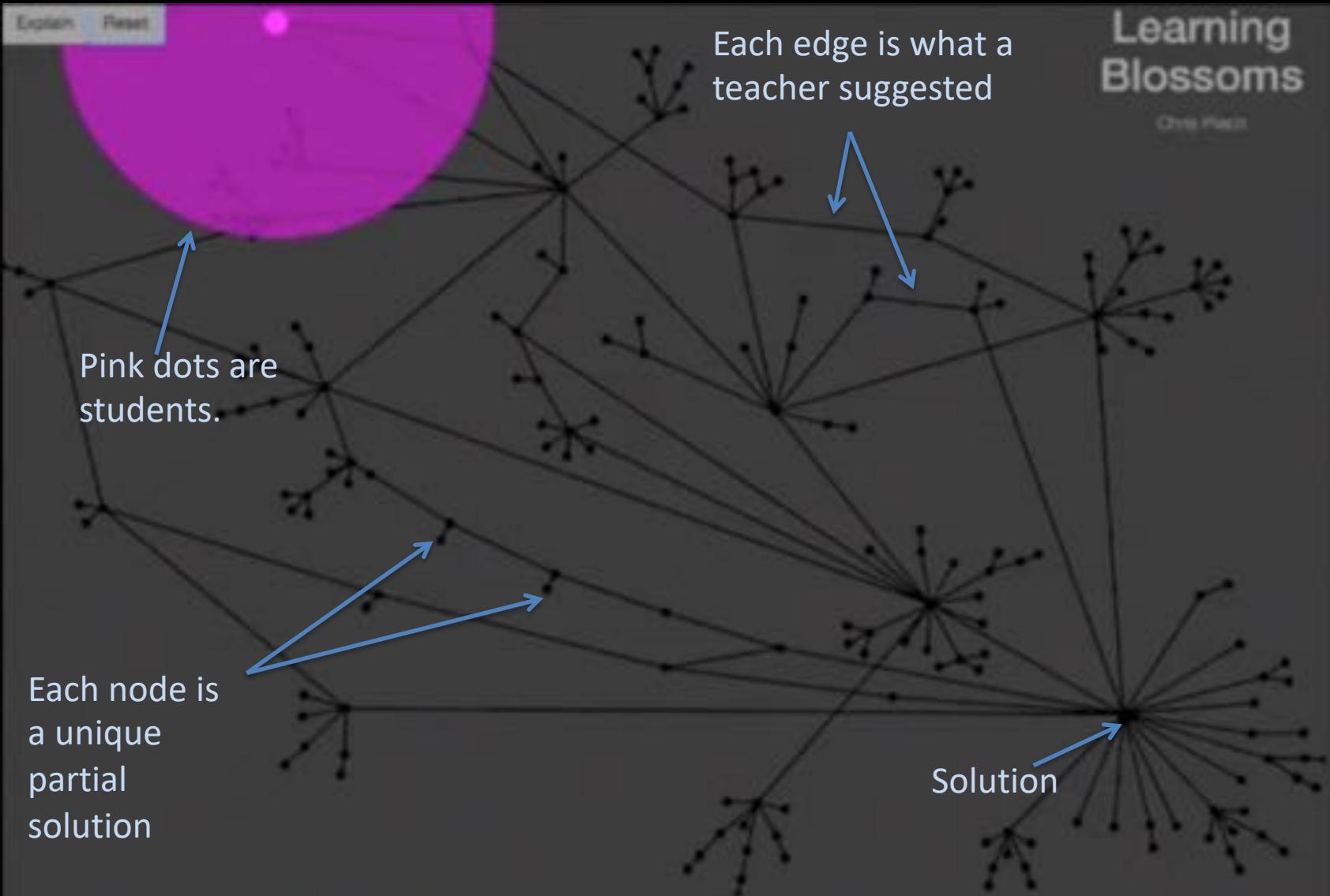
# **Chapter 1:** Better data source?

Explore Reset

# Learning Blossoms

Chris Pachl





Explain Reset

# Learning Blossoms

Chris Pach



# The Crowd is Un-wise

Temporal methods tried:

- Shortest path
- Min Time
- Expected Success
- Reinforcement learning
- Most Common Next
- Most Popular Path



when run  
move forward  
move forward  
turn left ⬅️⬇️

when run  
move forward  
turn left ⬅️⬇️

when run  
move forward  
move forward  
turn right ⬅️⬆️

when run  
move forward  
move forward  
turn left ⬅️⬇️  
move forward

18%

45%

12%



# Desirable Path Algorithms

## Poisson Common Path

$$\gamma(s) =$$



Predicted next  
partial solution

First step in the *most frequent* path to  
the solution from  $s$ , taken by *average*  
students. Assume poison process.



# Desirable Path Algorithms

Poisson Common Path

$$\gamma(s) = \arg \min_{p \in Z(s)} \sum_{x \in p} \frac{1}{\lambda_x}$$

Path Cost

Submission count of partial solution

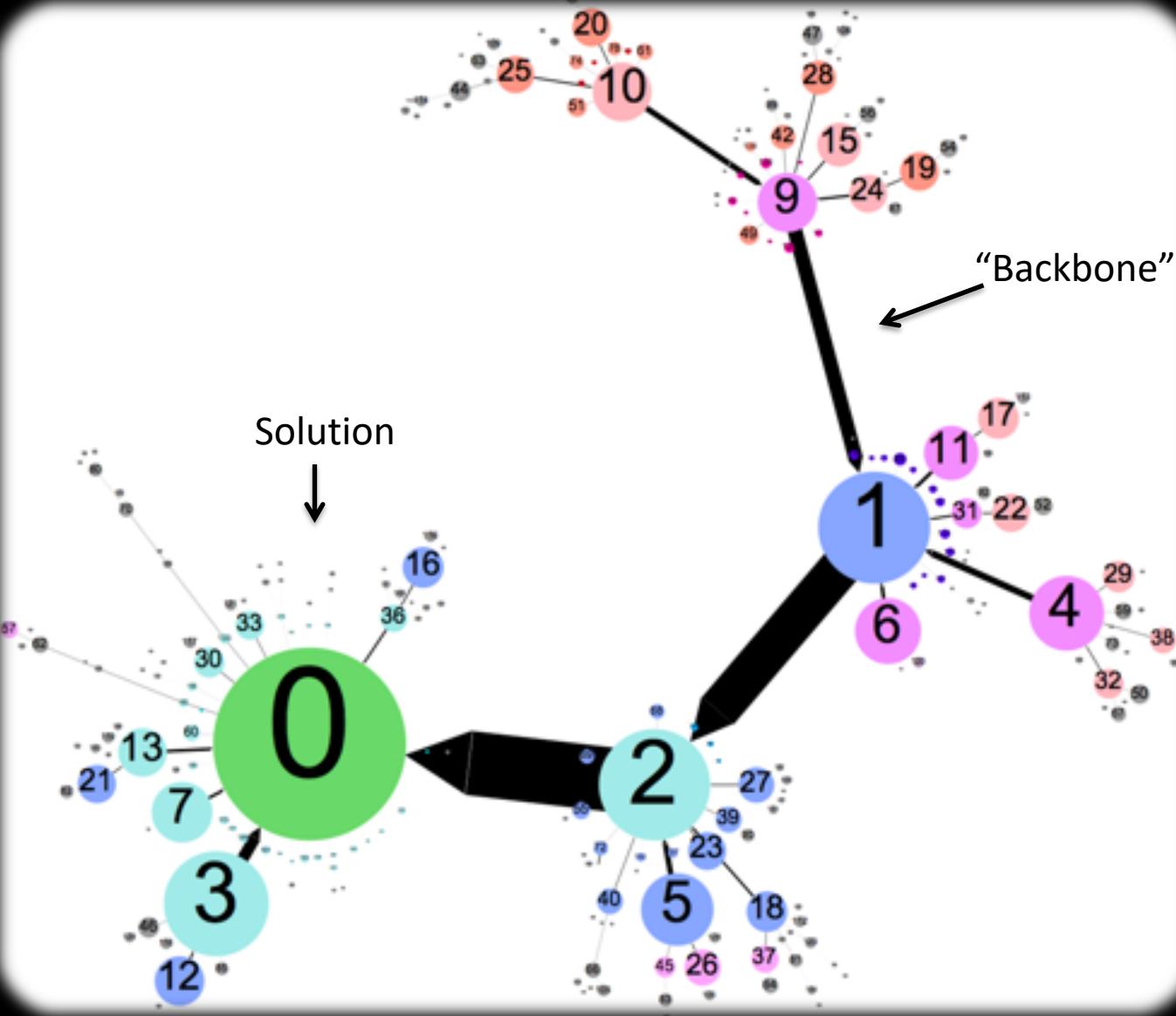
Predicted next partial solution

Paths to solution

Partial solutions in the path



# Learned Problem Solving Policy



Only worked well for 6 line programs...

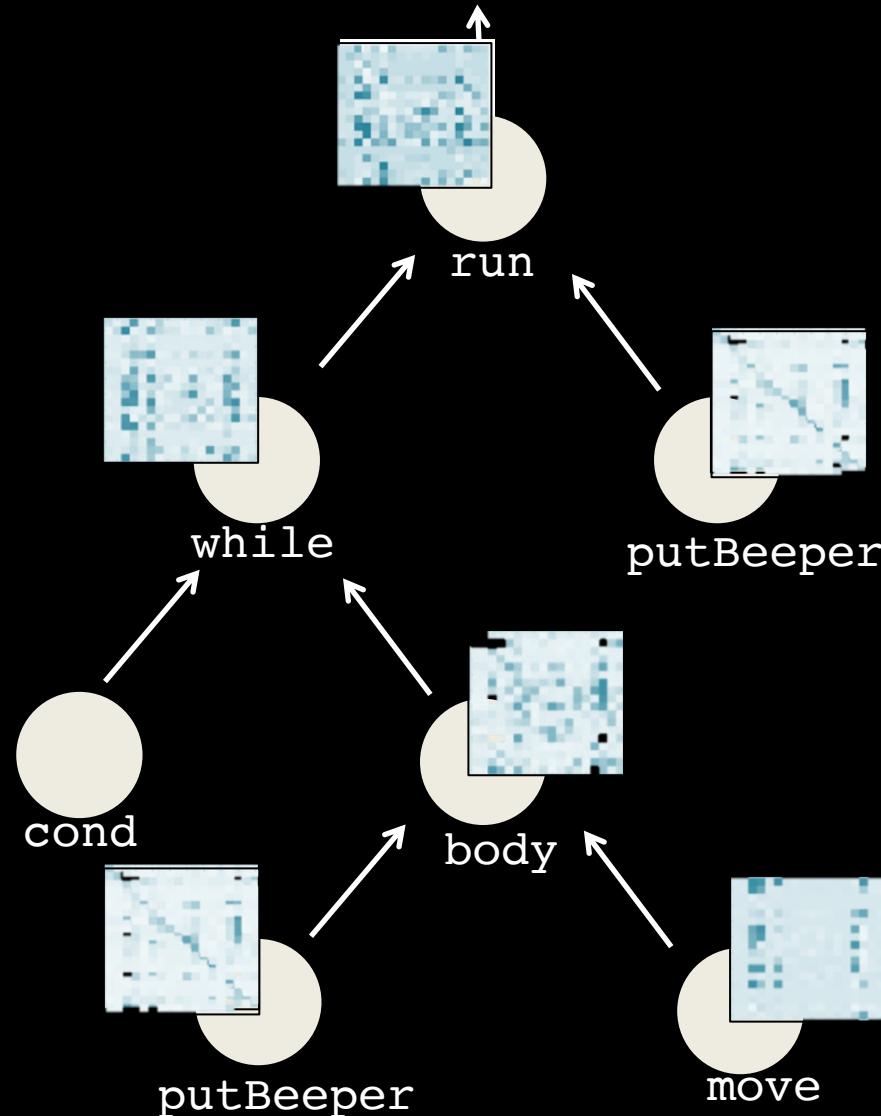
# **Chapter 2:** Start to invent new algorithms...



# Encode a Program

```
// User defined method  
private void run() {  
    while(isClear()){  
        putBeeperv());  
        move();  
    }  
    putBeeperv());  
}
```

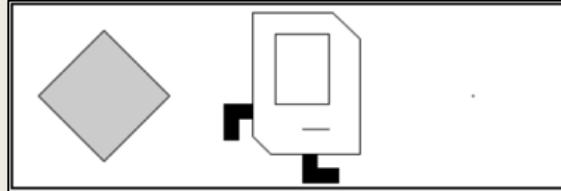
It looks like you have a fencepost error!



\*Note: this was coded pre-tensor flow



# Collect Triples



Precondition

---

```
putBeeper();  
move();
```

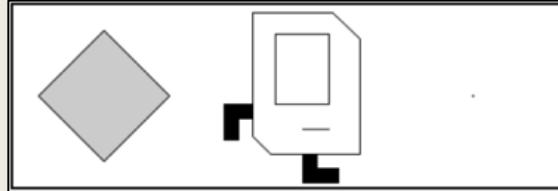
---

Code

About 5 million triples per assignment



# Collect Triples

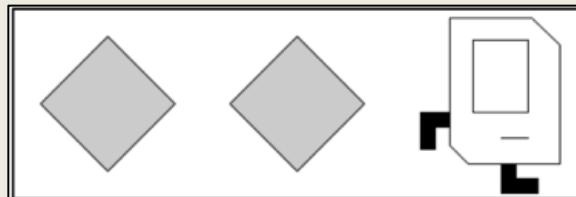


Precondition

---

```
putBeeper();  
move();
```

Code



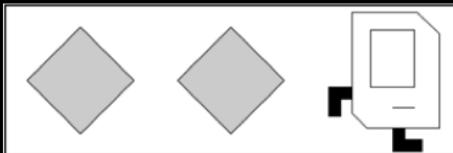
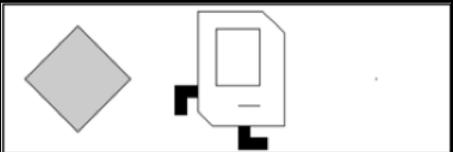
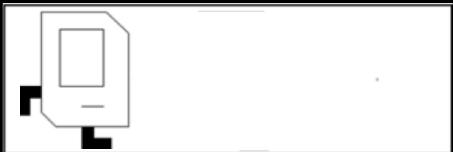
Postcondition

About 5 million triples per assignment

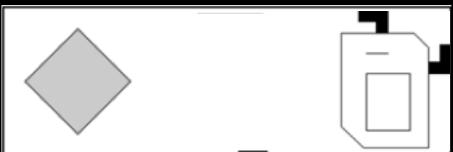


# A Code Phrase is a Mapping

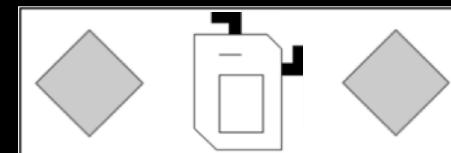
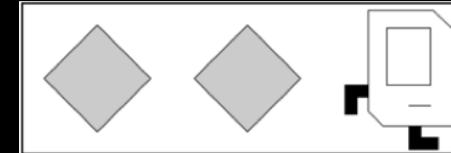
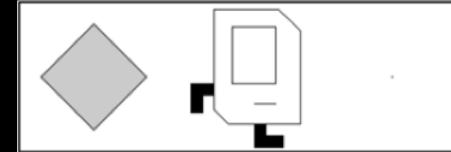
All possible preconditions



⋮



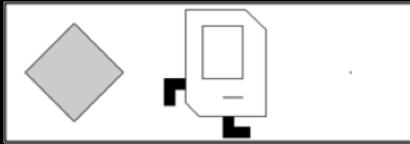
All possible postconditions



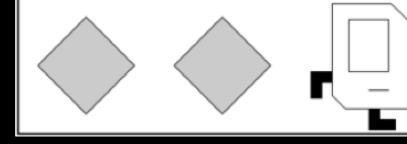
⋮



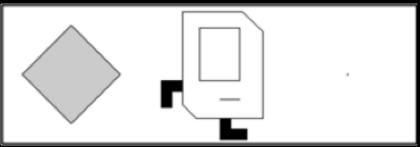
# Neural Network for Programs



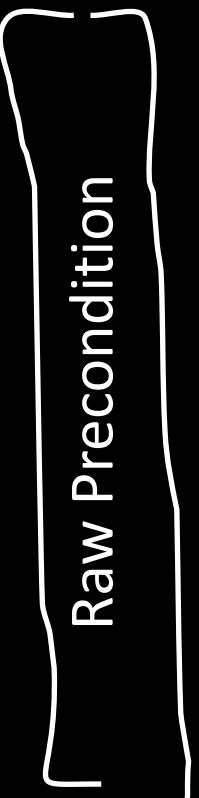
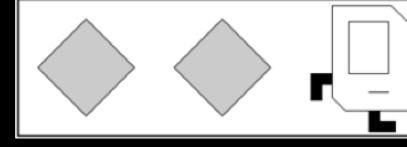
```
method step() {  
    putBeeper();  
    move();  
}
```



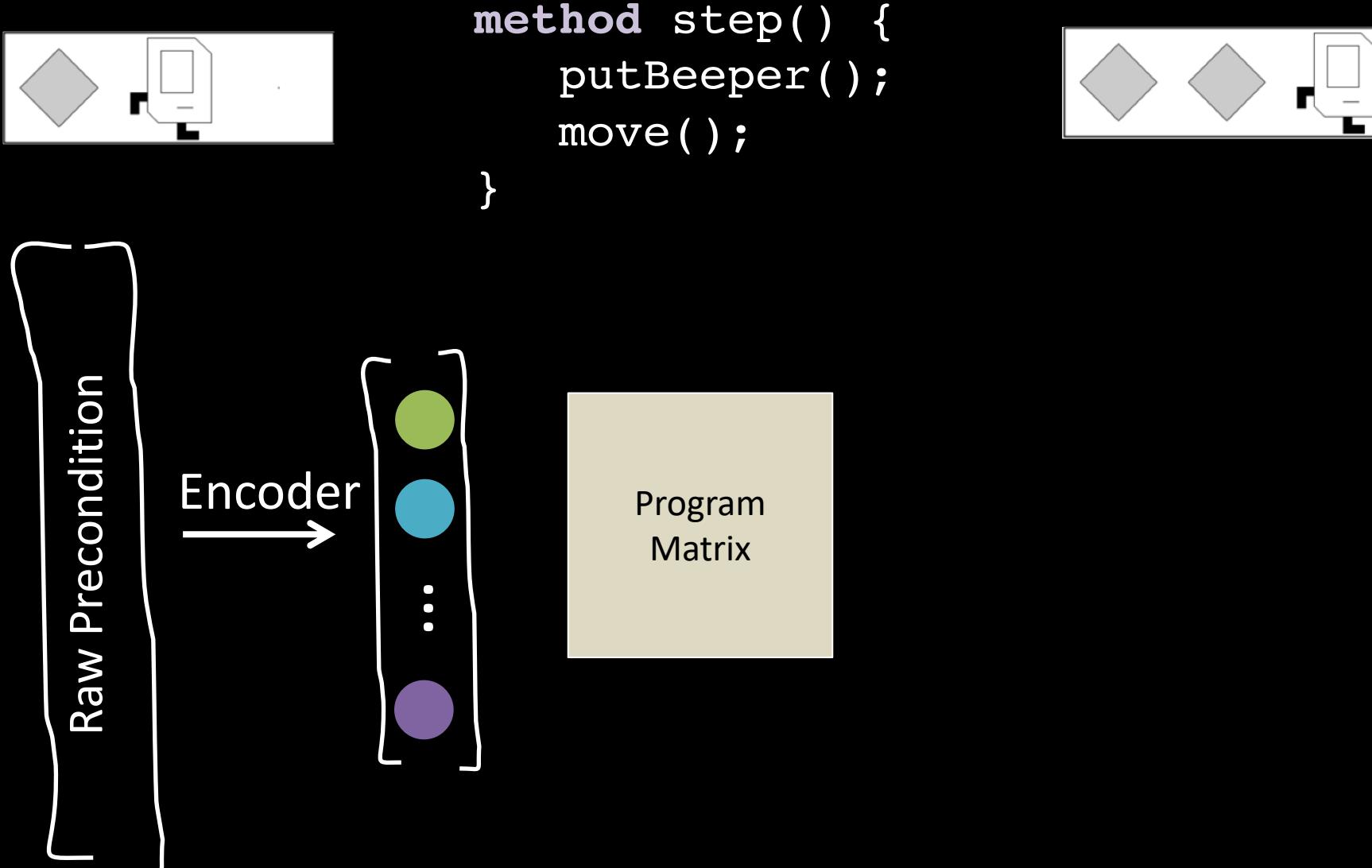
# Neural Network for Programs



```
method step() {  
    putBeeper();  
    move();  
}
```



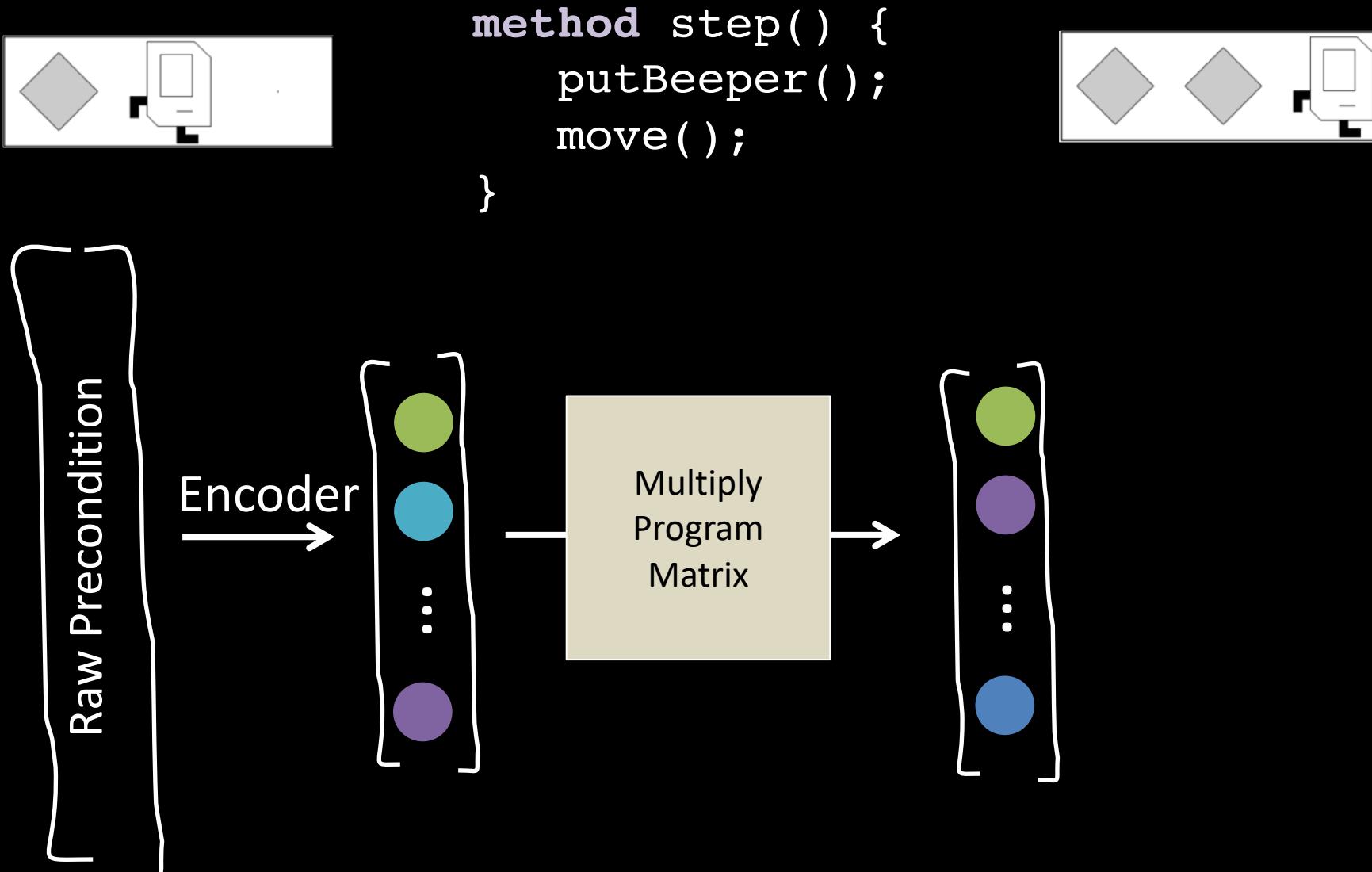
# Neural Network for Programs



\*coded pre-tensor flow



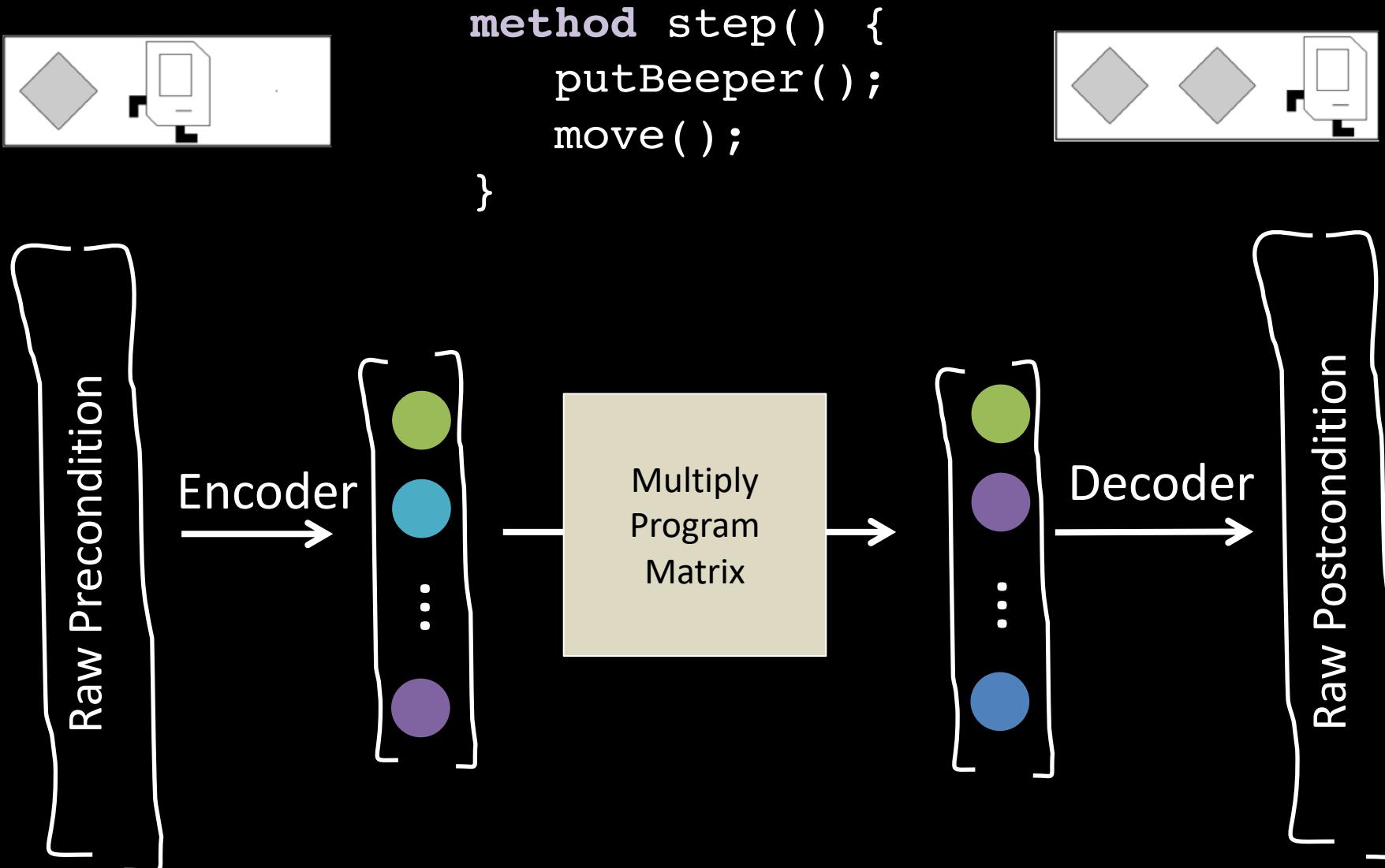
# Neural Network for Programs



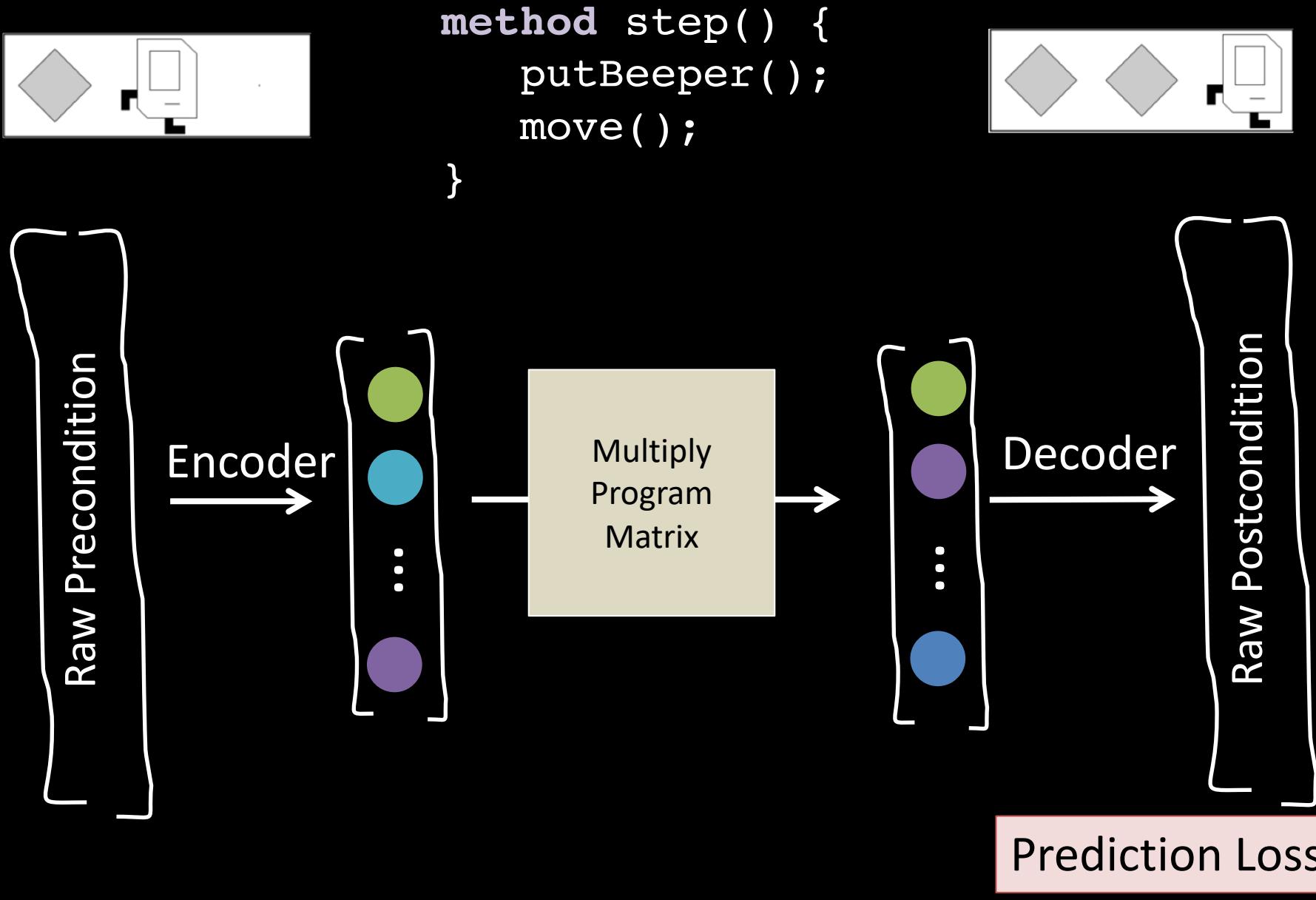
\*coded pre-tensor flow



# Neural Network for Programs



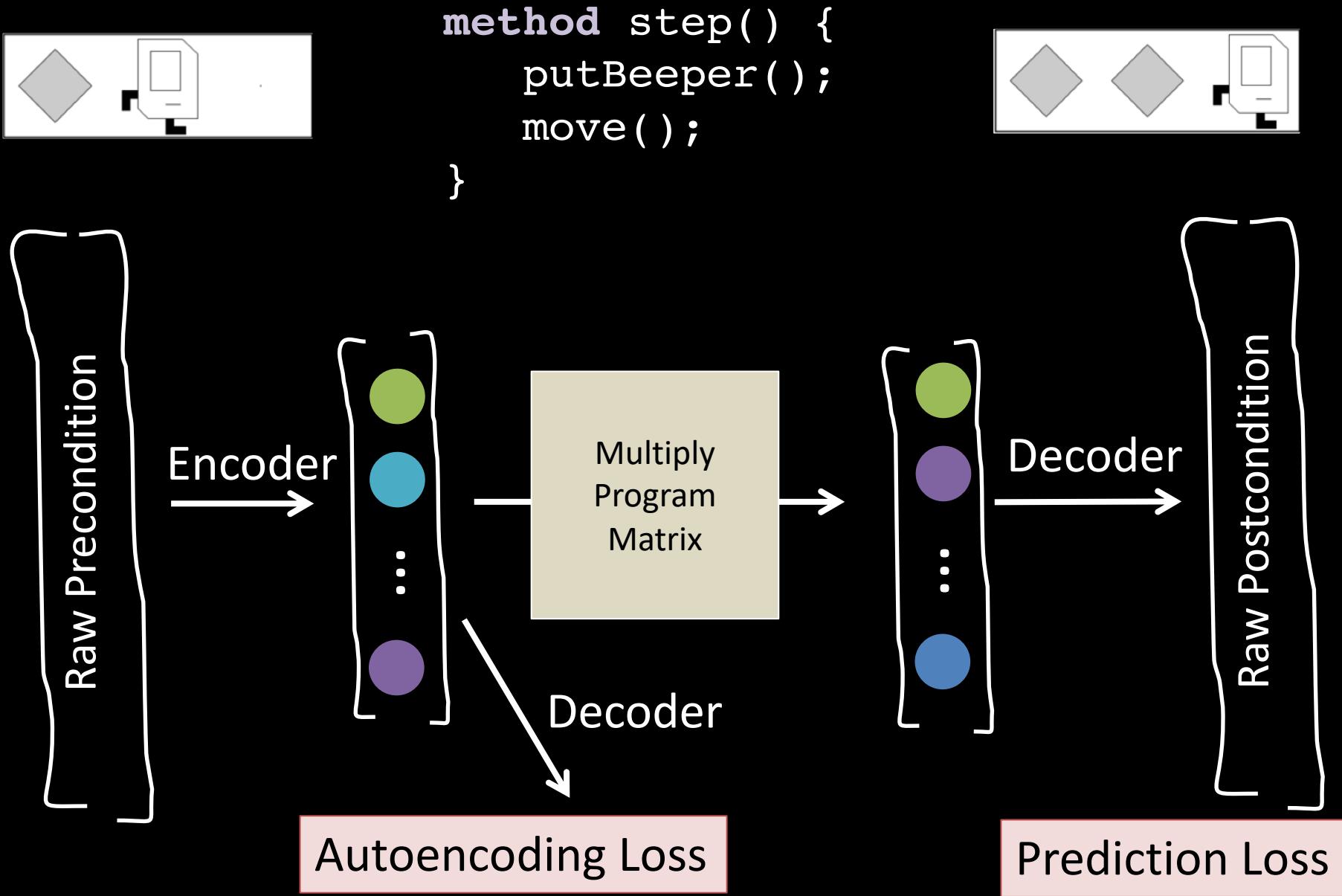
# Neural Network for Programs



\*coded pre-tensor flow



# Neural Network for Programs

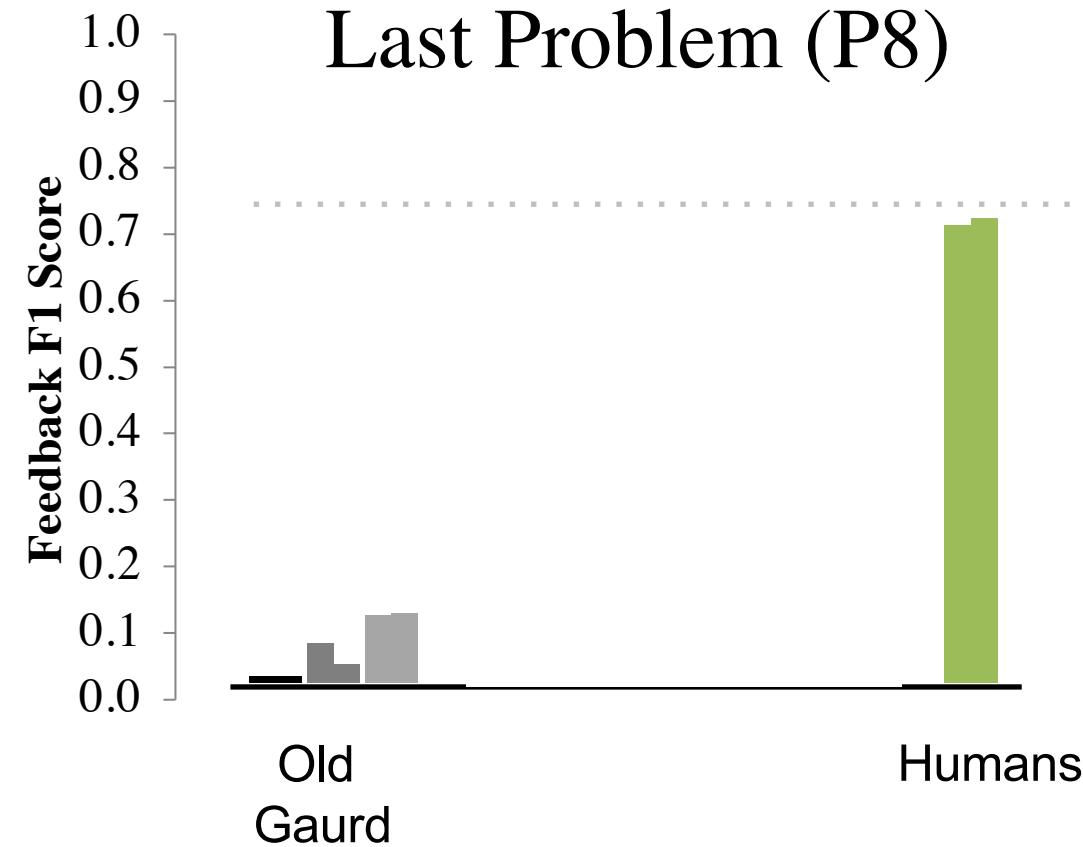


# Does it work?



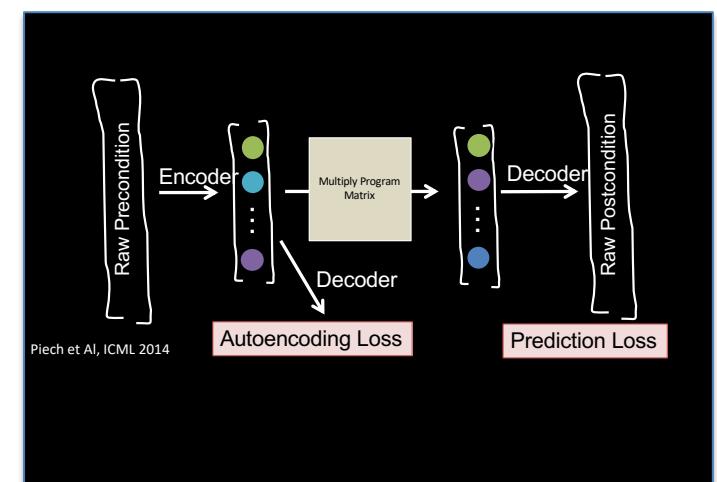
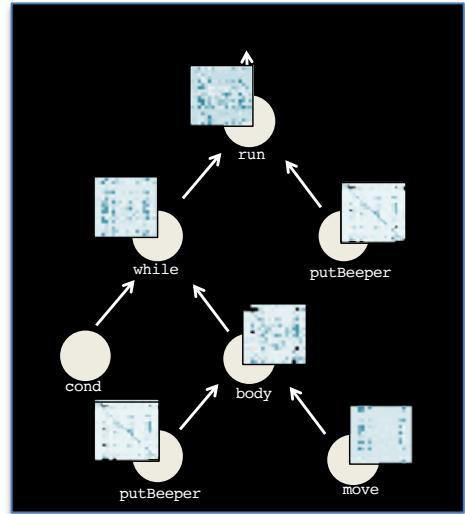
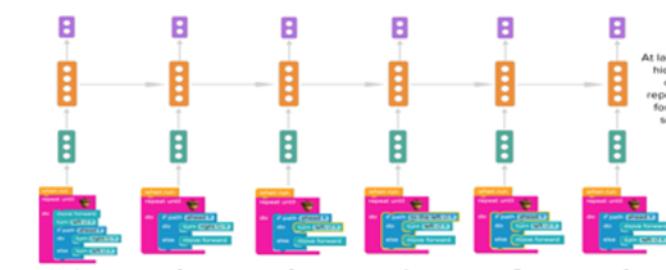
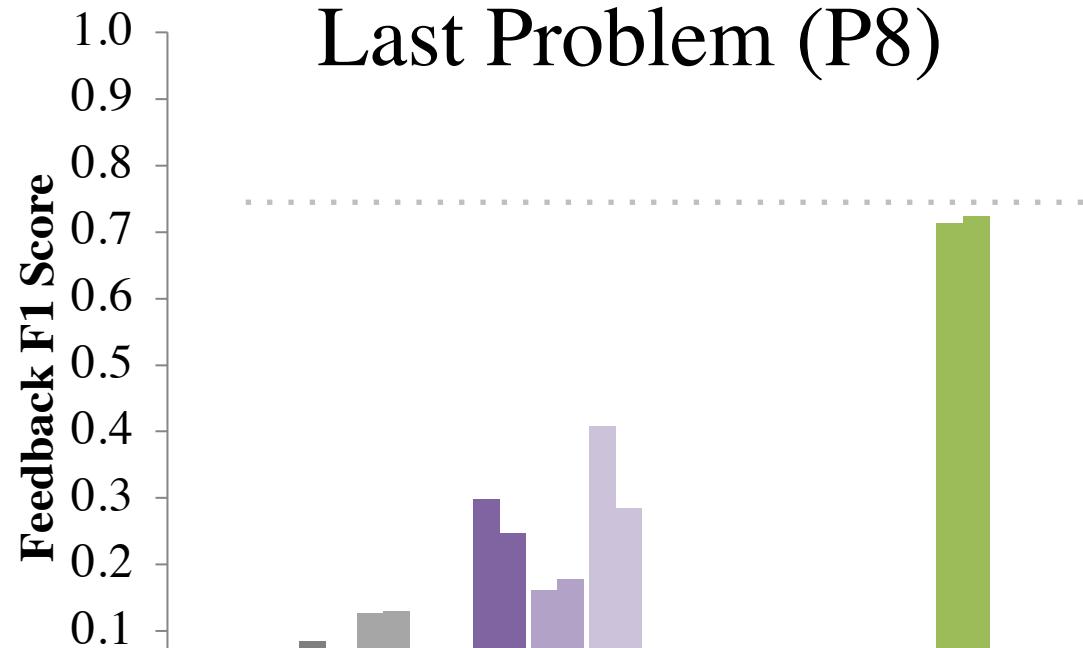
# Traditional Deep Learning Doesn't Work

Label student code



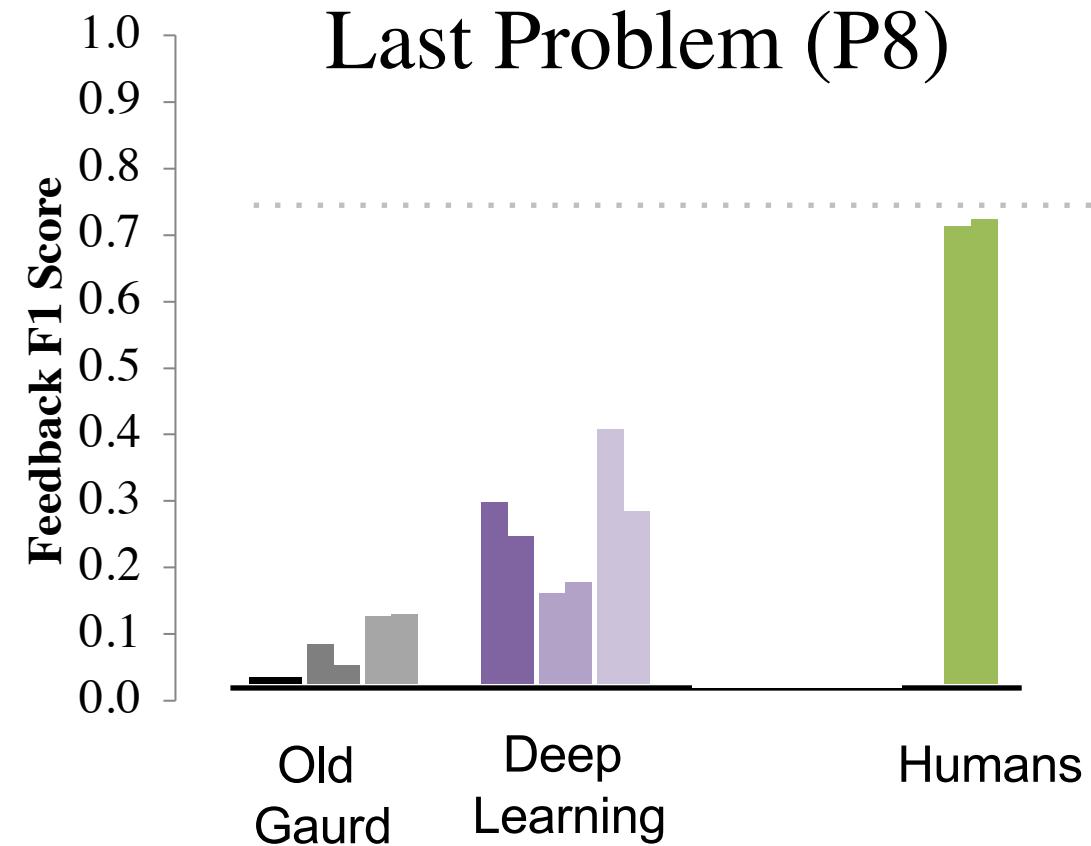
# Inaccurate, Uninterpretable, and Data Hungry

Label student code



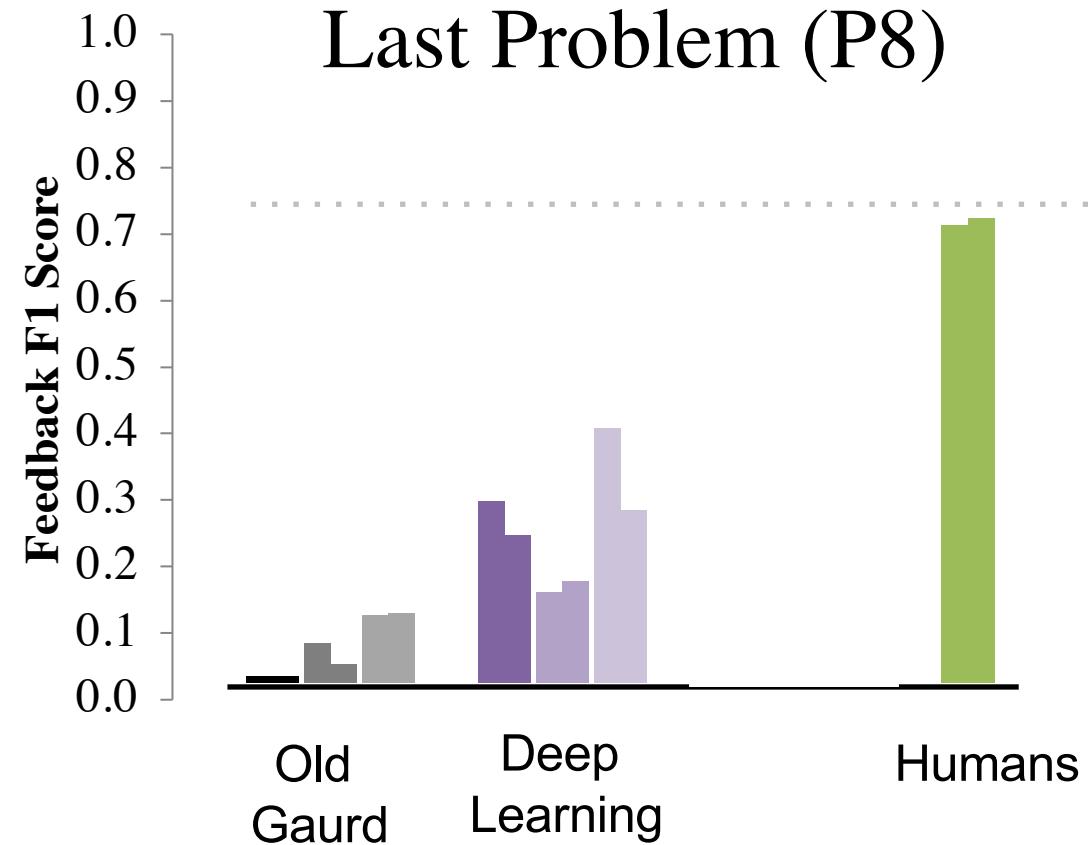
# Inaccurate, Uninterpretable, and Data Hungry

Label student code



# Data Hungry

Label student code



We need one  
shot learning

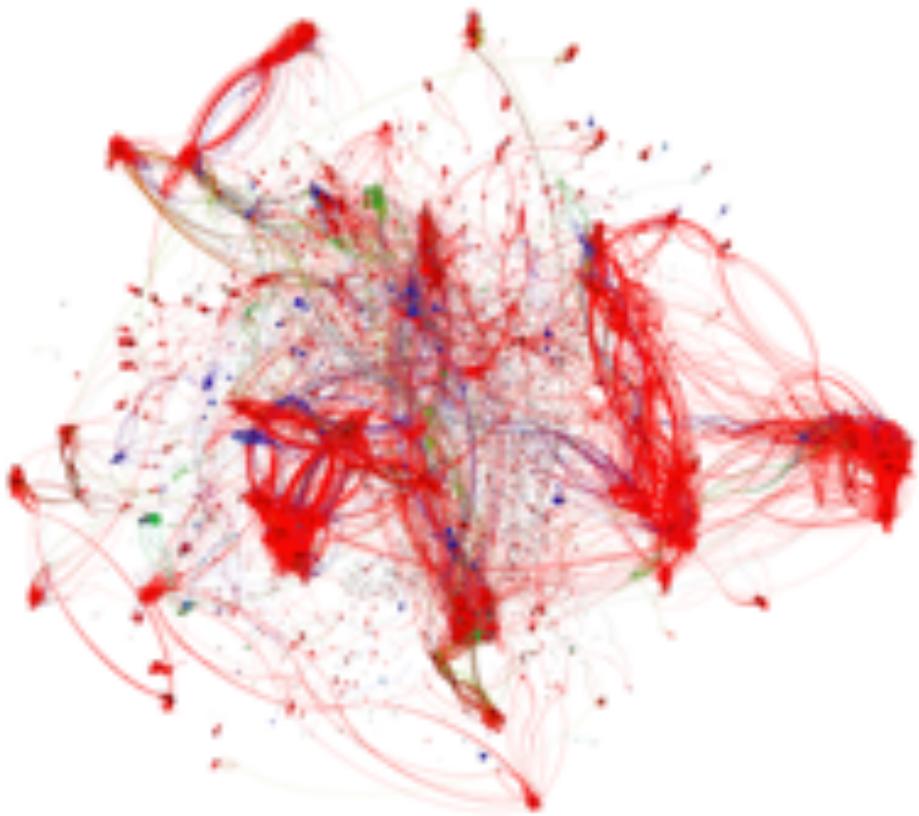
We need  
verifiability



Why is it so hard?

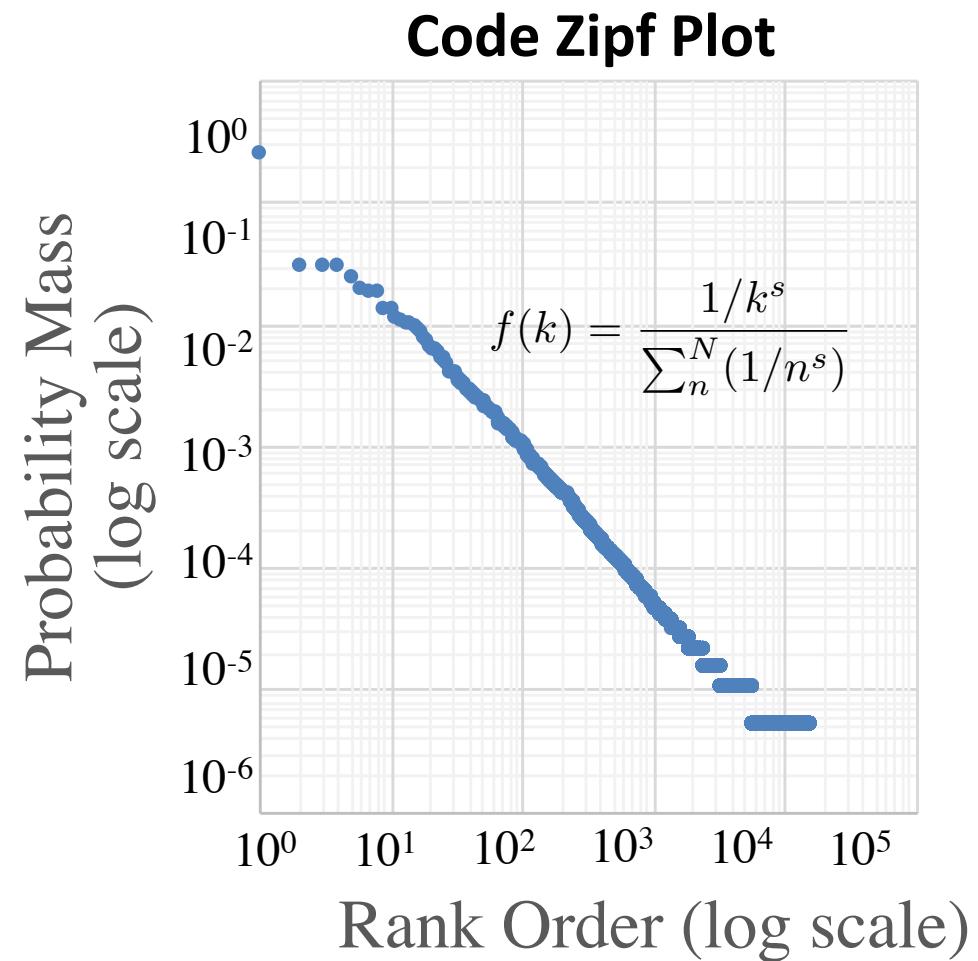
# Hard Problem

Brute force solution?



1 million unique solutions to  
programming Linear Regression

WWW 2014



[Suspense]

# **Chapter 3: Back to the drawing board**

# Humans Don't Need Much Data

Single training example:

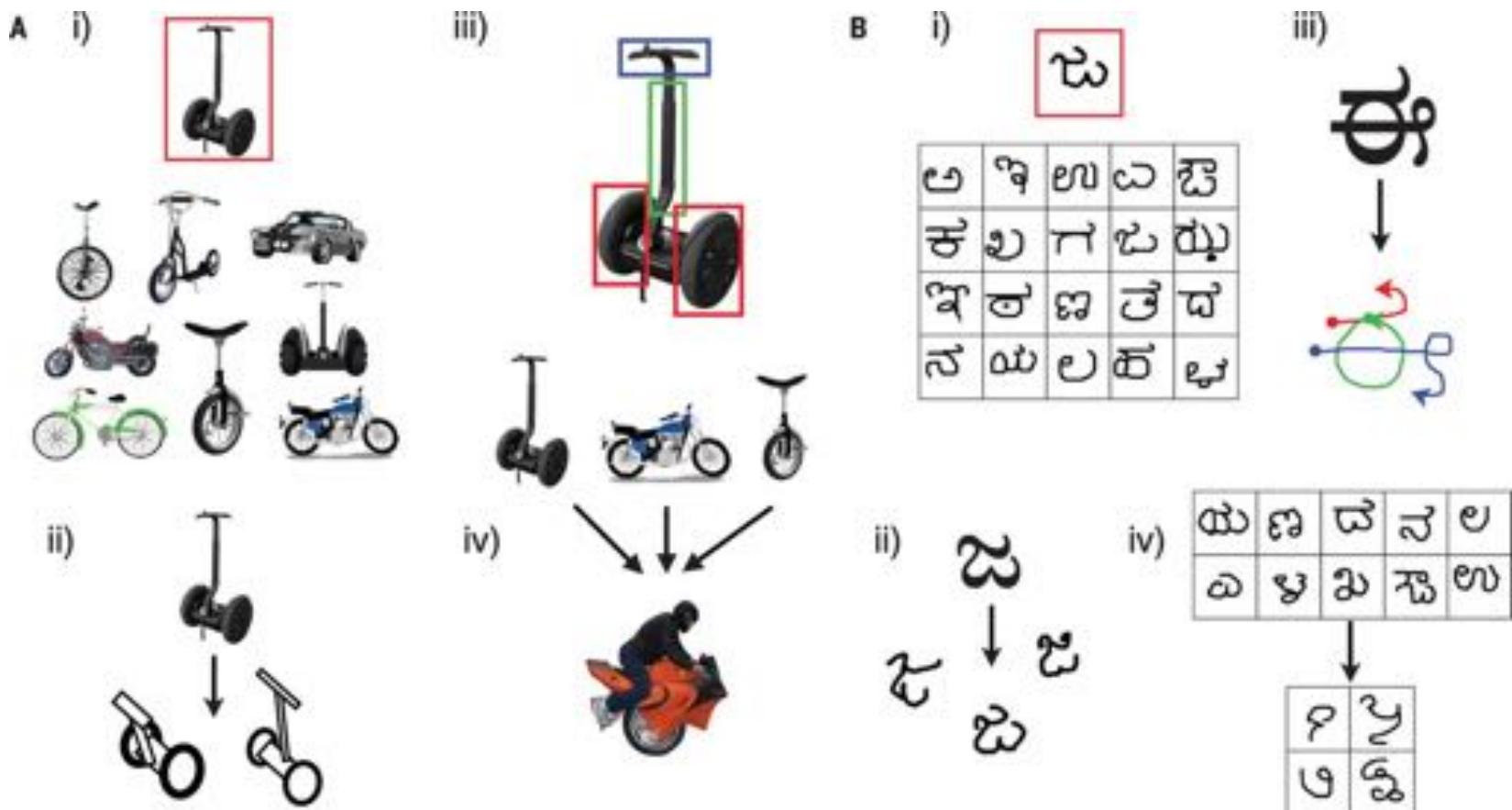
অ

Test set:

a অ ম স ম  
অ ম স ম  
আ ক ই এ  
অ ব ি এ  
আ ক ই এ



**Fig. 1 People can learn rich concepts from limited data.**



Brenden M. Lake et al. Science 2015;350:1332-1338

Copyright © 2015, American Association for the Advancement of Science

**Science**  
AAAS



**Fig. 2 Simple visual concepts for comparing human and machine learning.**

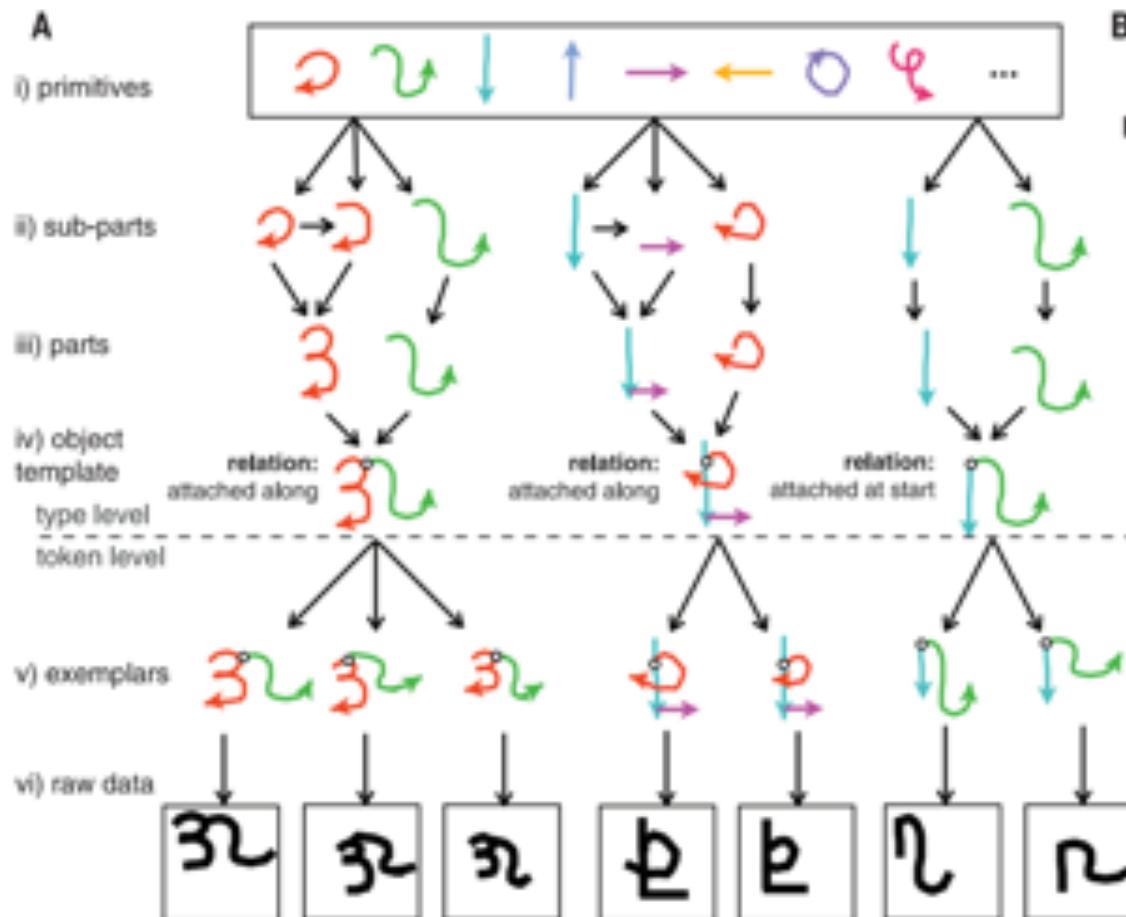


Brenden M. Lake et al. Science 2015;350:1332-1338

Copyright © 2015, American Association for the Advancement of Science

Science  
AAAS

# Bayesian Program Learning



**B**

```

procedure GENERATETYPE
   $\kappa \leftarrow P(\kappa)$                                 ▷ Sample number of parts
  for  $i = 1 \dots \kappa$  do
     $n_i \leftarrow P(n_i|\kappa)$                         ▷ Sample number of sub-parts
    for  $j = 1 \dots n_i$  do
       $s_{ij} \leftarrow P(s_{ij}|s_{i(j-1)})$             ▷ Sample sub-part sequence
    end for
     $R_i \leftarrow P(R_i|S_1, \dots, S_{i-1})$           ▷ Sample relation
  end for
   $\psi \leftarrow \{\kappa, R, S\}$ 
  return @GENERATETOKEN( $\psi$ )                      ▷ Return program

```

---

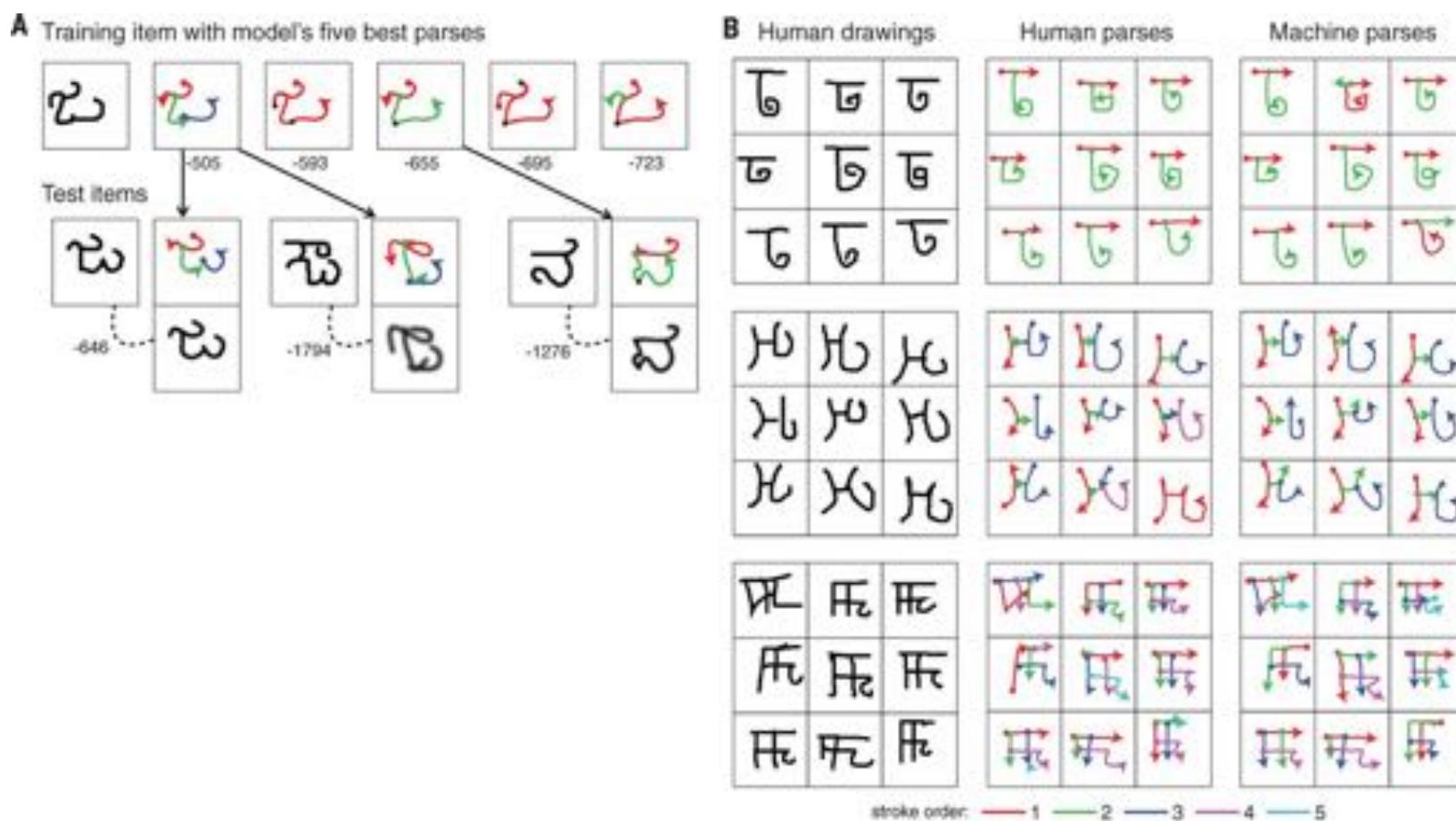
```

procedure GENERATETOKEN( $\psi$ )
  for  $i = 1 \dots \kappa$  do
     $S_i^{(m)} \leftarrow P(S_i^{(m)}|S_i)$                 ▷ Add motor variance
     $L_i^{(m)} \leftarrow P(L_i^{(m)}|R_i, T_1^{(m)}, \dots, T_{i-1}^{(m)})$ 
     $T_i^{(m)} \leftarrow f(L_i^{(m)}, S_i^{(m)})$            ▷ Sample part's start location
    Compose a part's trajectory
  end for
   $A^{(m)} \leftarrow P(A^{(m)})$                       ▷ Sample affine transform
   $I^{(m)} \leftarrow P(I^{(m)}|T^{(m)}, A^{(m)})$         ▷ Sample image
  return  $I^{(m)}$ 

```



**Fig. 4 Inferring motor programs from images.**



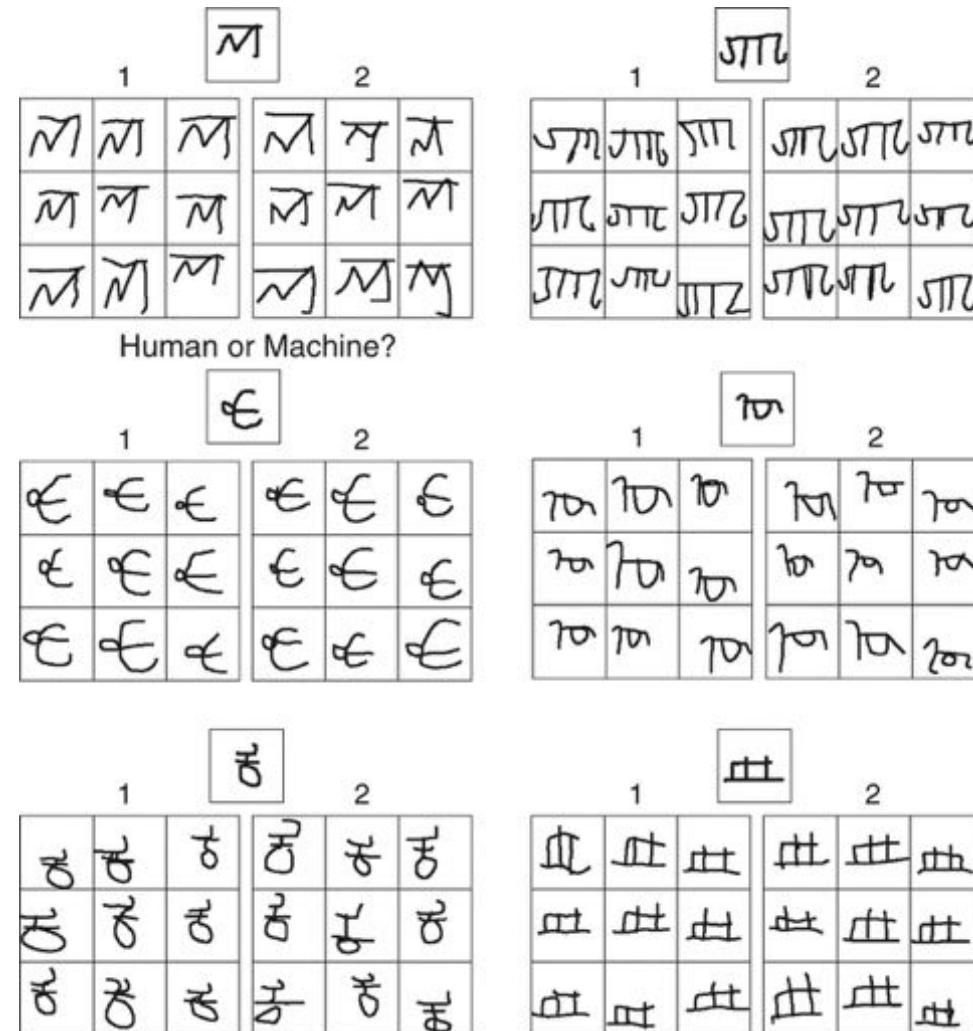
Brenden M. Lake et al. Science 2015;350:1332-1338

Copyright © 2015, American Association for the Advancement of Science

**Science**  
AAAS



**Fig. 5 Generating new exemplars.**



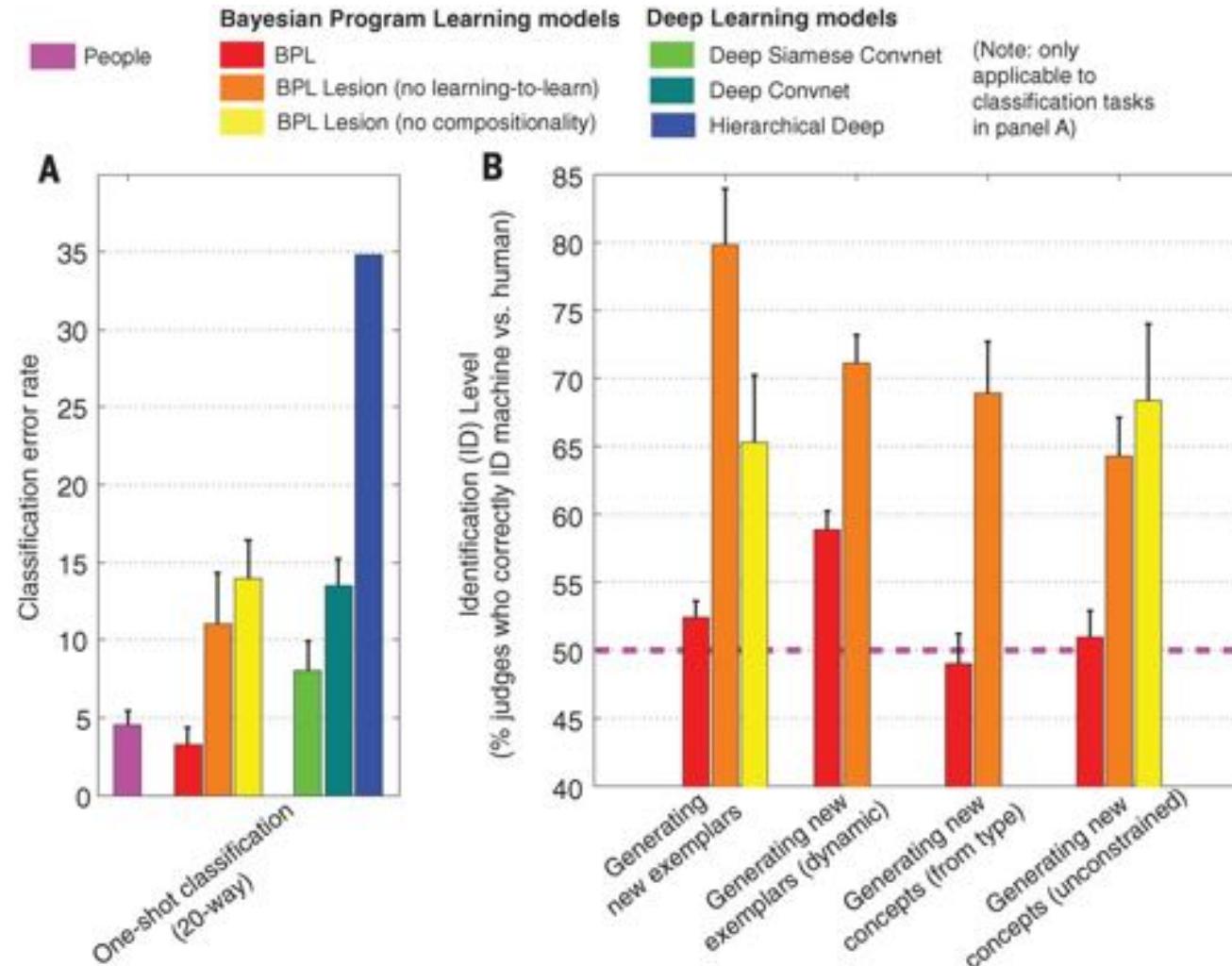
Brenden M. Lake et al. Science 2015;350:1332-1338



Copyright © 2015, American Association for the Advancement of Science

Science  
AAAS

**Fig. 6 Human and machine performance was compared on (A) one-shot classification and (B) four generative tasks.**



Brenden M. Lake et al. Science 2015;350:1332-1338

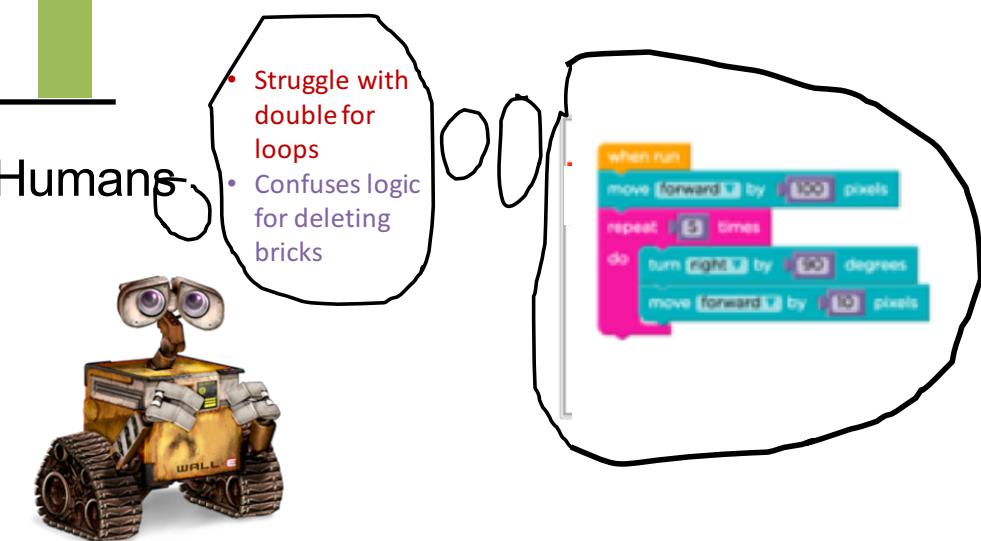
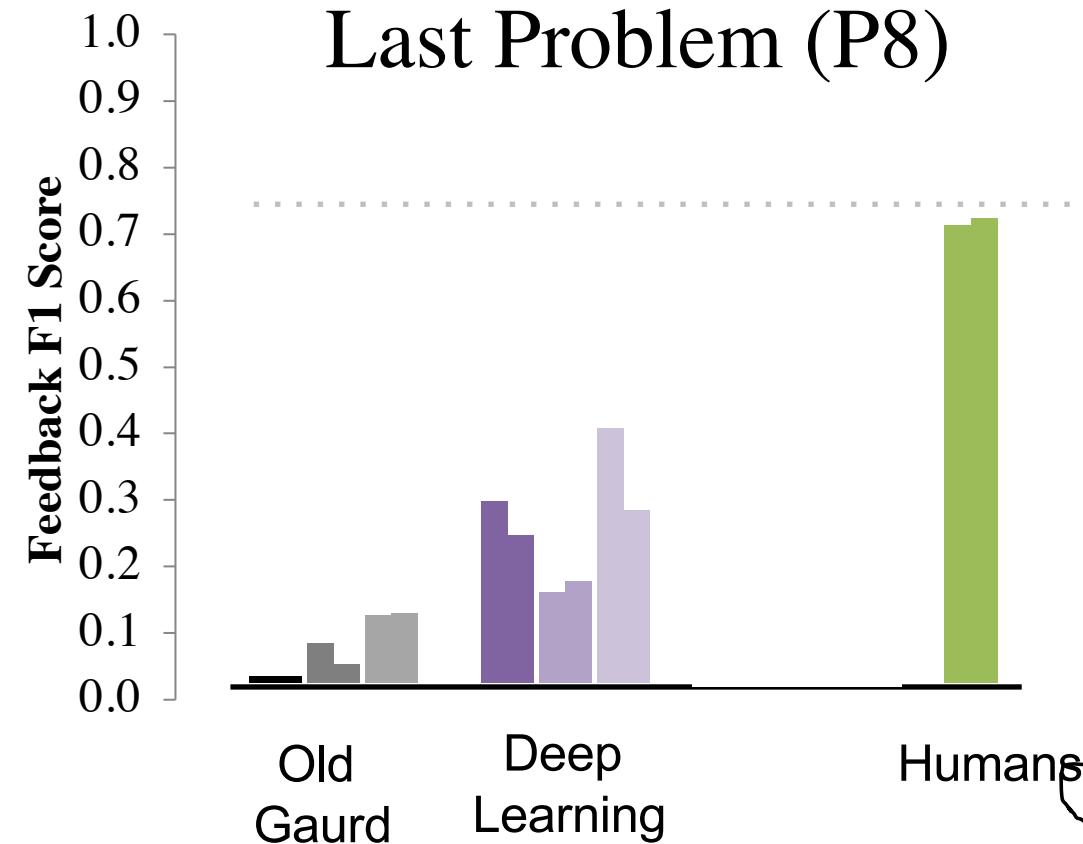
Copyright © 2015, American Association for the Advancement of Science

Science  
AAAS

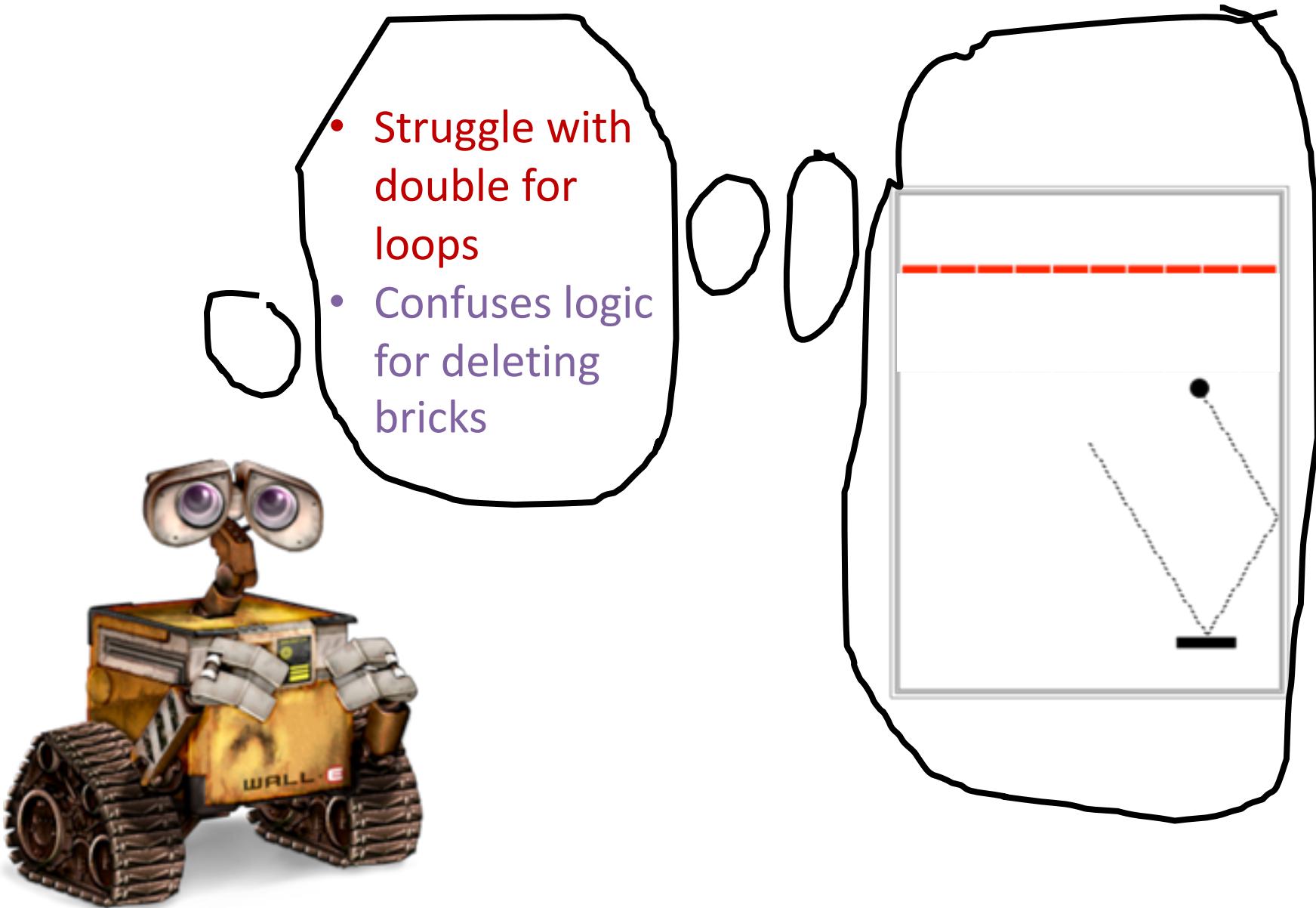


# Generative Understanding

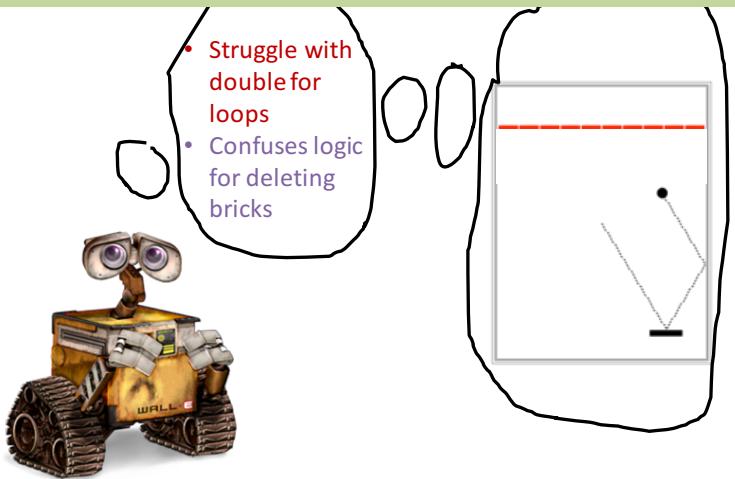
Label student code



# Imagine Students



# Imagine Students



This is easy and exponential



This is hard and linear

A students  
“ability”

$$\Theta \sim \text{pythonSample}$$

A students  
“choices”

$$C \sim \text{pythonSample} | \Theta$$

$$P(\Theta, C | \Pi)$$

Infer ability and  
choices from code

The resulting  
code

$$\Pi \sim \text{pythonSample} | C$$



# Bayesian Programming Language



ideaToText



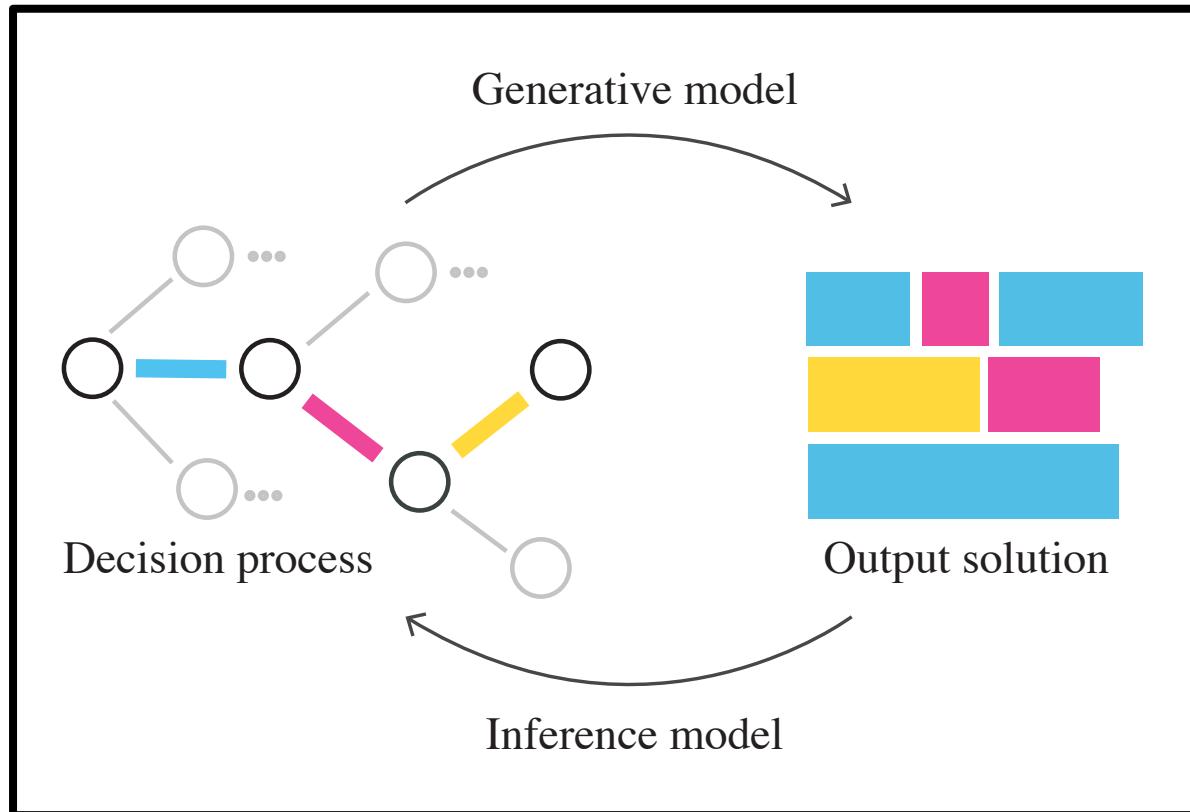
# Teachers Articulate N misconceptions

```
7 # This python class is a RubricSampling Decision
8 # it generates programs that print the numbers 10 -> 1
9
10 class Countdown(Decision):
11
12     def registerChoices(self):
13         # these are the main strategies for printing out a
14         # countdown
15         self.addRubricChoices('loop-style', {
16             'for' : θ1,
17             'while' : θ2,
18             'none' : θ3,
19             'empty' : θ4
20         })
21
22     # we can make some grading choices based on which
23     # strategy they chose (did they actually use a loop?)
24     def processChoices(self):
25         style = self.getChoice('loop-style')
26         hasLoop = style != 'none' and style != 'empty'
27         self.addLabel('rubric-hasLoop', hasLoop)
28
29
30     # Based on their strategy render a different decision
31     def renderCode(self):
32         style = self.getChoice('loop-style')
33         if style == 'for': return '{ForSoln}'
34         if style == 'while': return '{WhileSoln}'
35         if style == 'none': return '{NoLoopSoln}'
36         if style == 'empty': return ''
37
```

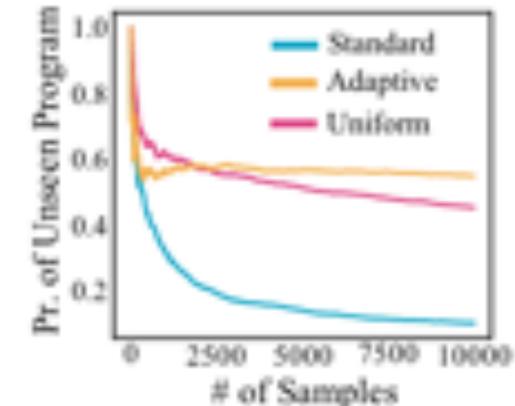
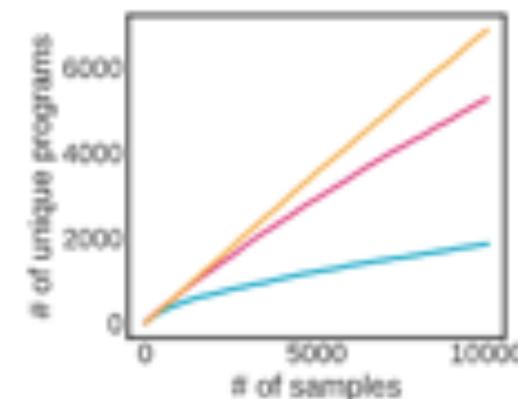
1. This is code for a single decision point
2. Give a name to the choice that the student is making
3. How do those choices translate into feedback?
4. What does the code look like? Often evokes other decision points



# Generative Understanding



Idea: (1) sample a ton, then (2) build a neural network to learn to predict decisions



Decision process

Next choice

$$p_{\mathcal{G}}(x_{a_1}, \dots, x_{a_T} | y) = \prod_{t=1}^T p_{\mathcal{G}}(x_{a_t} | y, \mathbf{x}_{<a_t})$$

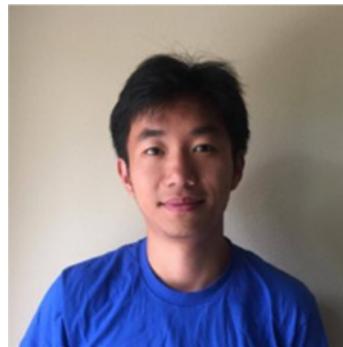
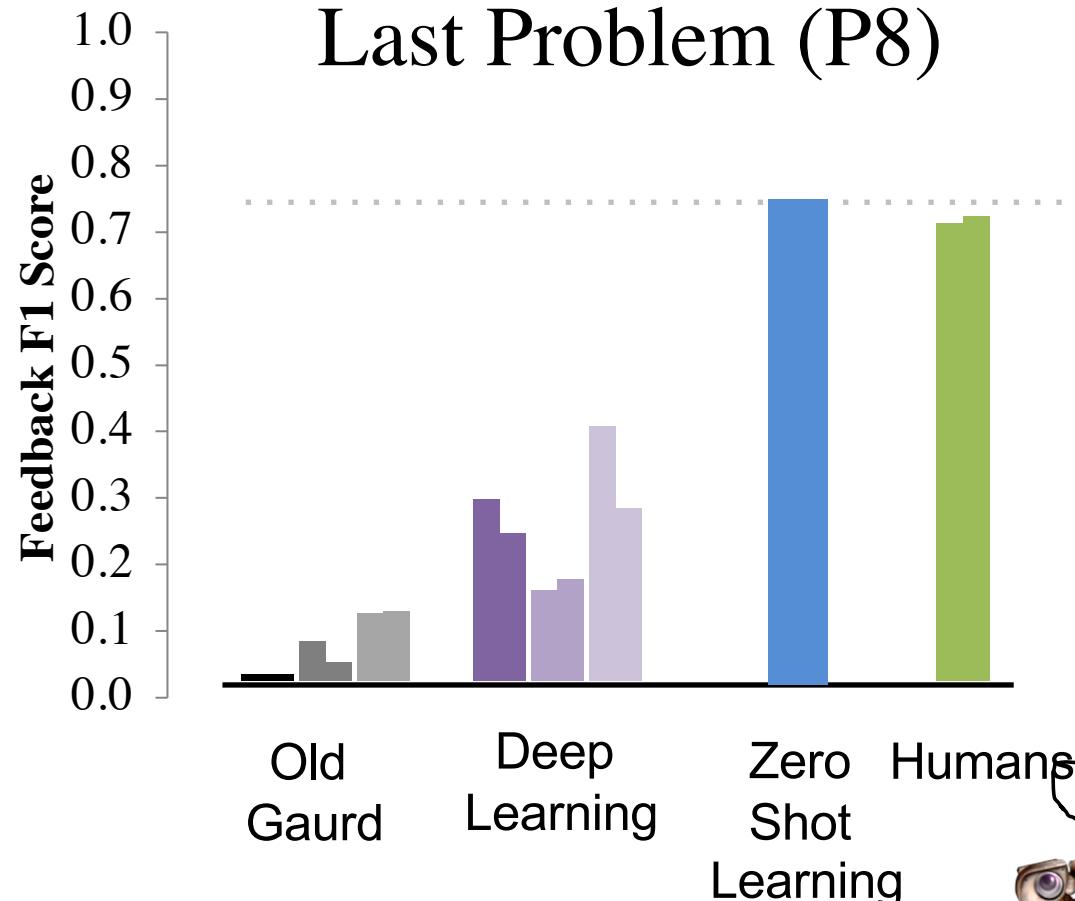
solution

Previous choices

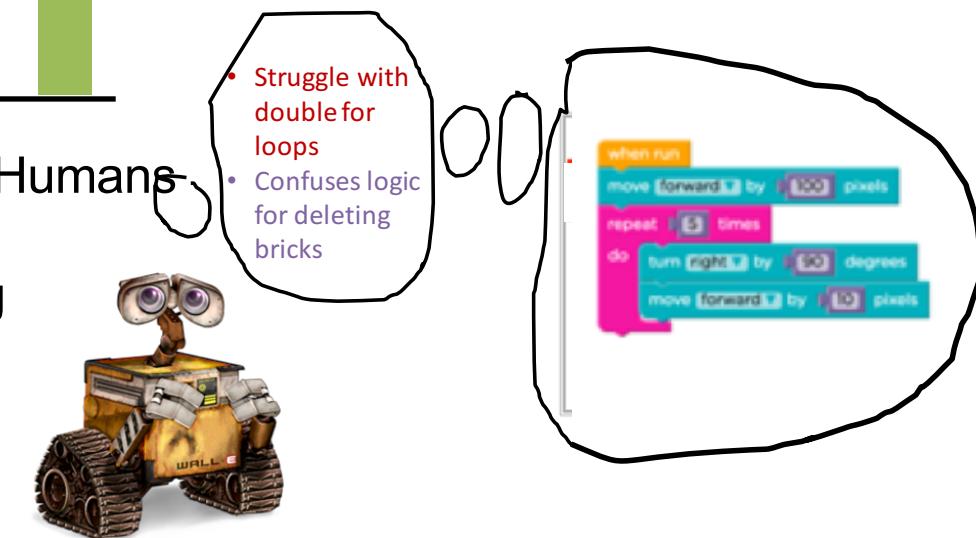


# Generative Understanding

Label student code



*Outstanding Student  
paper award, AAAI 2019*



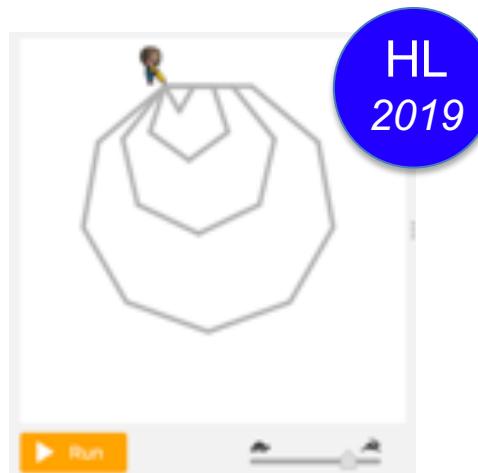
Not just for code

**Results from early 2019**

# Many domains of student work

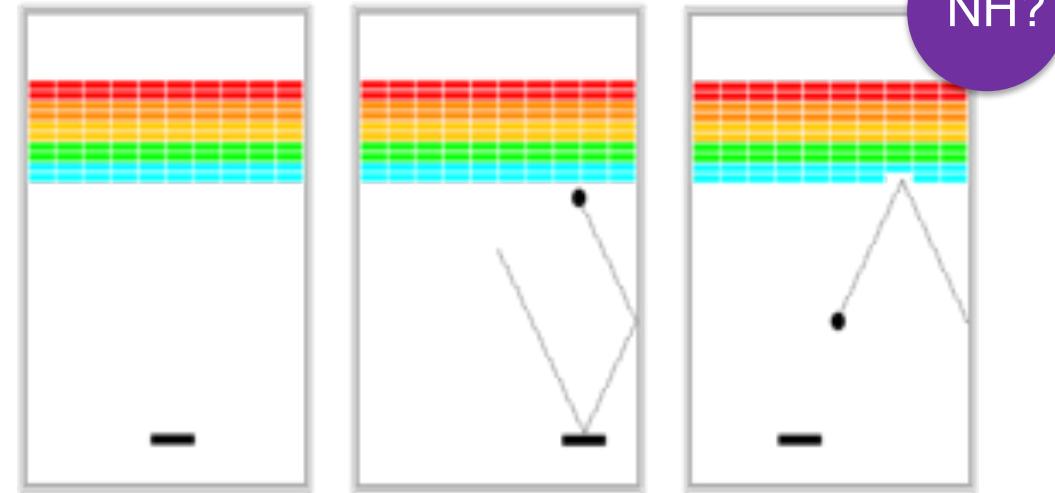
$$\begin{aligned} & 9 \times 6 \\ & = (10 - \square) \times 6 \\ & = 10 \times 6 - \square \times 6 \\ & = 60 - \square \\ & = \square \end{aligned}$$

HL  
2017



NH  
2019

Why did the original colonists come to America?



NH

Near Human

HL

Human Level

SH

Super Human Level

So what?

**What does this mean for me?!?**

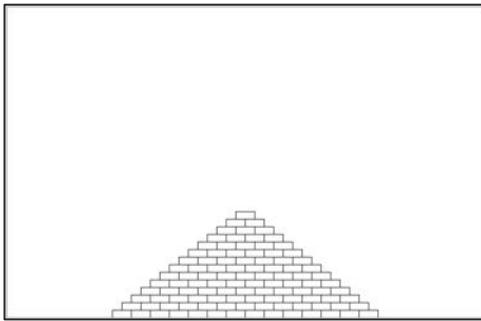
# Cutting Edge @ Stanford

2,600 students

130,000 partial solutions

$\mu$  snapshots per student = 50

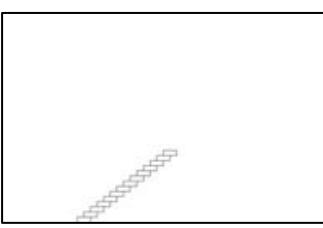
$\mu$  time per student = 2 hours



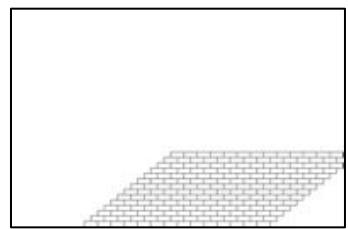
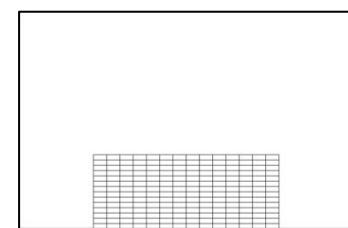
Step 1



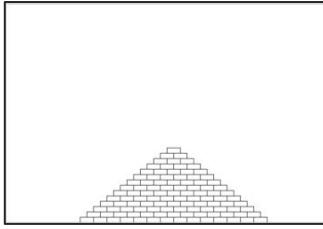
Step 2



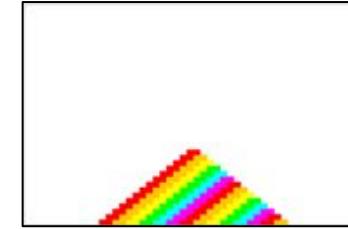
Step 3



Step 4



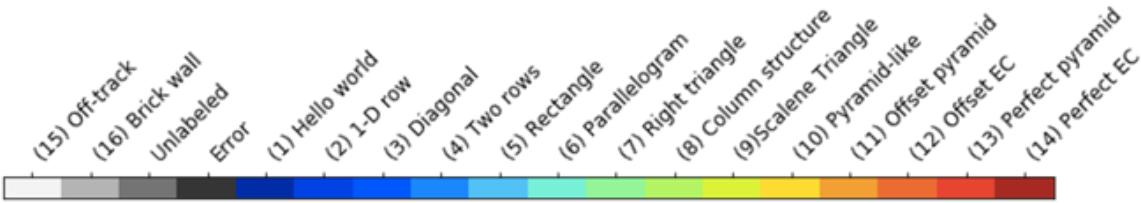
Step 5



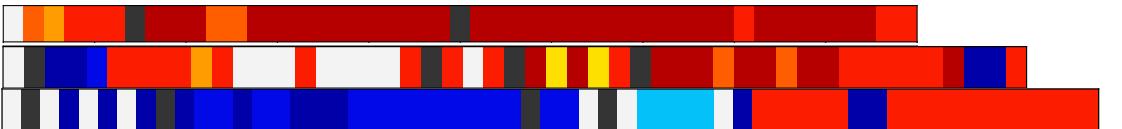
Step 6



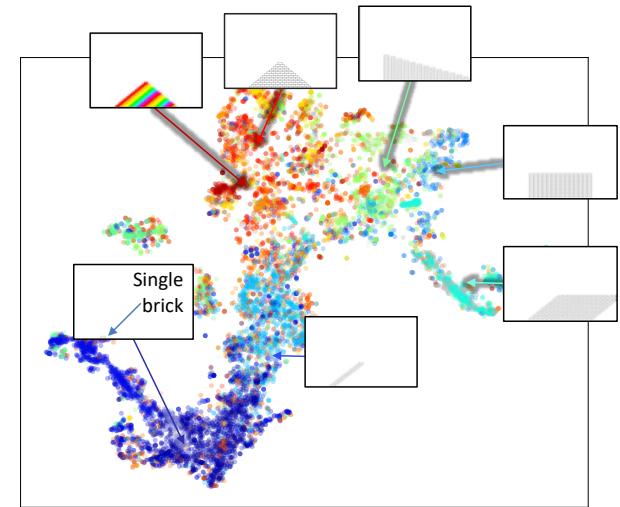
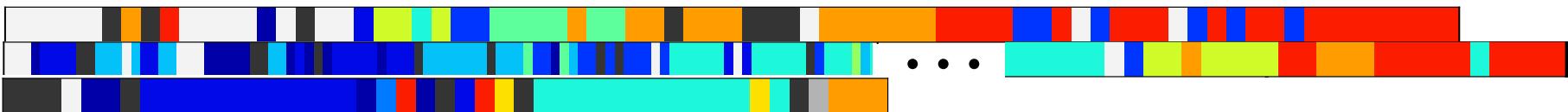
# Cutting Edge @ Stanford



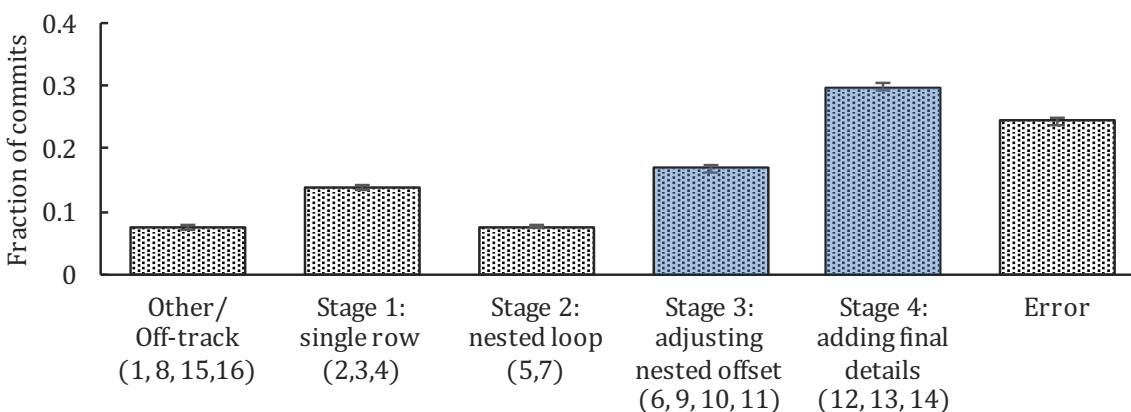
Students scoring in 99<sup>th</sup> percentile on midterm exam



Students scoring in  $\leq 3^{\text{rd}}$  percentile on midterm exam



t-SNE embedding of 130,000 partial solutions



# Cutting Edge @ Stanford

The screenshot shows a Java code editor with a file named `Pyramid.java`. The code is a Java program that generates a pyramid pattern using nested loops. A vertical timeline on the left indicates the progression of the assignment from 14 hours to 31 hours. A progress bar at the bottom shows the user has completed about 10% of the assignment. On the right, there is a graphical representation of the pyramid and a line graph titled "SourceLength" showing the number of characters in the source code over time. The graph tracks three metrics: Selection (green), Comments (orange), and Code (blue). The code length increases rapidly initially, then plateaus around 750 characters.

```
/*
 * File: Pyramid.java
 * Name: A.J. Alibana
 * Section Leader: Kaitlyn Legattuta
 *
 * This file is the starter file for the Pyramid problem.
 * It includes definitions of the constants that match the
 * sample run in the assignment, but you should make sure
 * that changing these values causes the generated display
 * to change accordingly.
 */
import acm.graphics.*;
import acm.program.*;
import java.awt.*;

public class Pyramid extends GraphicsProgram {
    /** Width of each brick in pixels */
    private static final int BRICK_WIDTH = 30;

    /** Height of each brick in pixels */
    private static final int BRICK_HEIGHT = 12;

    /** Number of bricks in the base of the pyramid */
    private static final int BRICKS_IN_BASE = 18;

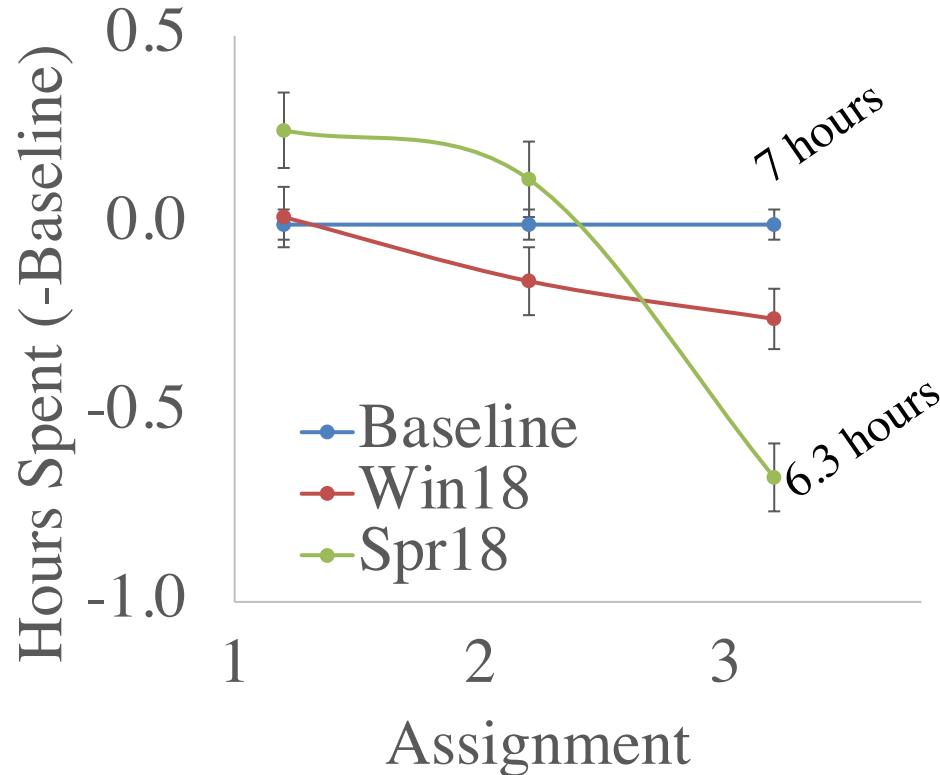
    public void run() {
        int x = (getWidth() / 2) - (BRICKS_IN_BASE / 2) * BRICK_WIDTH;
        int y = getHeight() - BRICK_HEIGHT;
        for(int row = BRICKS_IN_BASE; row > 0; row--) {
            layBricks(row, x, y);
            y += BRICK_HEIGHT;
            x += BRICK_WIDTH / 2;
        }
    }
}
```

SourceLength

Time into Problem (hours)	Selection	Comments	Code
0.0	100	100	100
0.1	150	150	150
0.2	200	200	200
0.3	250	250	250
0.4	300	300	300
0.5	350	350	350
0.6	400	400	400
0.7	450	450	450
0.8	500	500	500
0.9	550	550	550
1.0	600	600	600
1.1	650	650	650
1.2	700	700	700
1.3	750	750	750
1.4	750	750	750
1.5	750	750	750
1.6	750	750	750
1.7	750	750	750
1.8	750	750	750
1.9	750	750	750
2.0	750	750	750
2.1	750	750	750
2.2	750	750	750
2.3	750	750	750
2.4	750	750	750
2.5	750	750	750
2.6	750	750	750
2.7	750	750	750
2.8	750	750	750
2.9	750	750	750
3.0	750	750	750
3.1	750	750	750
3.2	750	750	750
3.3	750	750	750
3.4	750	750	750
3.5	750	750	750
3.6	750	750	750
3.7	750	750	750
3.8	750	750	750
3.9	750	750	750
4.0	750	750	750
4.1	750	750	750
4.2	750	750	750
4.3	750	750	750
4.4	750	750	750
4.5	750	750	750
4.6	750	750	750
4.7	750	750	750
4.8	750	750	750
4.9	750	750	750
5.0	750	750	750
5.1	750	750	750
5.2	750	750	750
5.3	750	750	750
5.4	750	750	750
5.5	750	750	750
5.6	750	750	750
5.7	750	750	750
5.8	750	750	750
5.9	750	750	750
6.0	750	750	750
6.1	750	750	750
6.2	750	750	750
6.3	750	750	750
6.4	750	750	750
6.5	750	750	750
6.6	750	750	750
6.7	750	750	750
6.8	750	750	750
6.9	750	750	750
7.0	750	750	750
7.1	750	750	750
7.2	750	750	750
7.3	750	750	750
7.4	750	750	750
7.5	750	750	750
7.6	750	750	750
7.7	750	750	750
7.8	750	750	750
7.9	750	750	750
8.0	750	750	750
8.1	750	750	750
8.2	750	750	750
8.3	750	750	750
8.4	750	750	750
8.5	750	750	750
8.6	750	750	750
8.7	750	750	750
8.8	750	750	750
8.9	750	750	750
9.0	750	750	750
9.1	750	750	750
9.2	750	750	750
9.3	750	750	750
9.4	750	750	750
9.5	750	750	750
9.6	750	750	750
9.7	750	750	750
9.8	750	750	750
9.9	750	750	750
10.0	750	750	750
10.1	750	750	750
10.2	750	750	750
10.3	750	750	750
10.4	750	750	750
10.5	750	750	750
10.6	750	750	750
10.7	750	750	750
10.8	750	750	750
10.9	750	750	750
11.0	750	750	750
11.1	750	750	750
11.2	750	750	750
11.3	750	750	750
11.4	750	750	750
11.5	750	750	750
11.6	750	750	750
11.7	750	750	750
11.8	750	750	750
11.9	750	750	750
12.0	750	750	750
12.1	750	750	750
12.2	750	750	750
12.3	750	750	750
12.4	750	750	750
12.5	750	750	750
12.6	750	750	750
12.7	750	750	750
12.8	750	750	750
12.9	750	750	750
13.0	750	750	750
13.1	750	750	750
13.2	750	750	750
13.3	750	750	750
13.4	750	750	750
13.5	750	750	750
13.6	750	750	750
13.7	750	750	750
13.8	750	750	750
13.9	750	750	750
14.0	750	750	750
14.1	750	750	750
14.2	750	750	750
14.3	750	750	750
14.4	750	750	750
14.5	750	750	750
14.6	750	750	750
14.7	750	750	750
14.8	750	750	750
14.9	750	750	750
15.0	750	750	750
15.1	750	750	750
15.2	750	750	750
15.3	750	750	750
15.4	750	750	750
15.5	750	750	750
15.6	750	750	750
15.7	750	750	750
15.8	750	750	750
15.9	750	750	750
16.0	750	750	750
16.1	750	750	750
16.2	750	750	750
16.3	750	750	750
16.4	750	750	750
16.5	750	750	750
16.6	750	750	750
16.7	750	750	750
16.8	750	750	750
16.9	750	750	750
17.0	750	750	750
17.1	750	750	750
17.2	750	750	750
17.3	750	750	750
17.4	750	750	750
17.5	750	750	750
17.6	750	750	750
17.7	750	750	750
17.8	750	750	750
17.9	750	750	750
18.0	750	750	750
18.1	750	750	750
18.2	750	750	750
18.3	750	750	750
18.4	750	750	750
18.5	750	750	750
18.6	750	750	750
18.7	750	750	750
18.8	750	750	750
18.9	750	750	750
19.0	750	750	750
19.1	750	750	750
19.2	750	750	750
19.3	750	750	750
19.4	750	750	750
19.5	750	750	750
19.6	750	750	750
19.7	750	750	750
19.8	750	750	750
19.9	750	750	750
20.0	750	750	750
20.1	750	750	750
20.2	750	750	750
20.3	750	750	750
20.4	750	750	750
20.5	750	750	750
20.6	750	750	750
20.7	750	750	750
20.8	750	750	750
20.9	750	750	750
21.0	750	750	750
21.1	750	750	750
21.2	750	750	750
21.3	750	750	750
21.4	750	750	750
21.5	750	750	750
21.6	750	750	750
21.7	750	750	750
21.8	750	750	750
21.9	750	750	750
22.0	750	750	750
22.1	750	750	750
22.2	750	750	750
22.3	750	750	750
22.4	750	750	750
22.5	750	750	750
22.6	750	750	750
22.7	750	750	750
22.8	750	750	750
22.9	750	750	750
23.0	750	750	750
23.1	750	750	750
23.2	750	750	750
23.3	750	750	750
23.4	750	750	750
23.5	750	750	750
23.6	750	750	750
23.7	750	750	750
23.8	750	750	750
23.9	750	750	750
24.0	750	750	750
24.1	750	750	750
24.2	750	750	750
24.3	750	750	750
24.4	750	750	750
24.5	750	750	750
24.6	750	750	750
24.7	750	750	750
24.8	750	750	750
24.9	750	750	750
25.0	750	750	750
25.1	750	750	750
25.2	750	750	750
25.3	750	750	750
25.4	750	750	750
25.5	750	750	750
25.6	750	750	750
25.7	750	750	750
25.8	750	750	750
25.9	750	750	750
26.0	750	750	750
26.1	750	750	750
26.2	750	750	750
26.3	750	750	750
26.4	750	750	750
26.5	750	750	750
26.6	750	750	750
26.7	750	750	750
26.8	750	750	750
26.9	750	750	750
27.0	750	750	750
27.1	750	750	750
27.2	750	750	750
27.3	750	750	750
27.4	750	750	750
27.5	750	750	750
27.6	750	750	750
27.7	750	750	750
27.8	750	750	750
27.9	750	750	750
28.0	750	750	750
28.1	750	750	750
28.2	750	750	750
28.3	750	750	750
28.4	750	750	750
28.5	750	750	750
28.6	750	750	750
28.7	750		

# Cutting Edge @ Stanford

Using assignment *timing* as pre-post



$$E[\hat{X}_3|X_1] - E[X_3] = 42 \text{ mins}$$

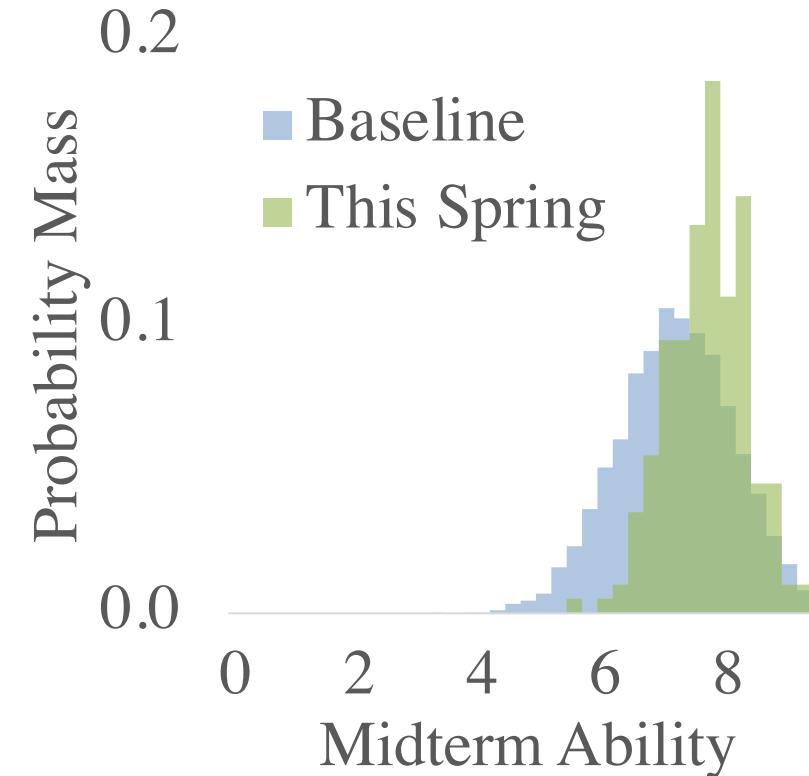
Predicted time

Actual time

$p < 0.00001$

Assignments are taking less time

Item Response Theory based ability assessment



$$S_{i,j} = n \cdot \sigma(a_i - d_j)$$

Score  
points

ability

difficulty

Students perform better than expected



Next trial  
**Code in Place 2021**

But its not perfect

**Hit a ceiling with code > 20 lines**

# PurpleBook

Browse Q3    DynamoDB - AWS Console

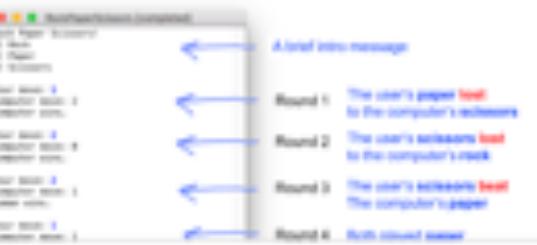
localhost:3000/#/view/cs106aMidWin19/3/abigail/cplech/bdaa714e79a10cb

Question   Solution   Starter   <

## Rock Paper Scissors (30 points)



Write a `ConsoleProgram` that has a user play rock paper scissors against a computer until either the user or the computer has three "wins". To make the code simpler, use integers to represent the different plays (0 is rock, 1 is paper, 2 is scissors). Example run:



Student	Answer
	<pre>1 public class RockPaperScissors extends ConsoleProgram { 2 3     /* constants */ 4     private static final int ROCK = 0; 5     private static final int PAPER = 1; 6     private static final int SCISSORS = 2; 7     private static final int N_WINS = 3; 8 9     private RandomGenerator rg = new RandomGenerator(); 10 11    public void run() { 12        introMessage(); 13        for (int i = 0; i &lt; N_WINS; i++) { 14            inputNumber(); 15            roundWinner(); 16        } 17        gameWinner(); 18    } 19 20    private void introMessage() { 21        println("Rock Paper Scissors!"); 22        println("0) Rock"); 23        println("1) Paper"); 24        println("2) Scissors"); 25        println(" "); 26    } 27 28    private void inputNumber() { 29        int a = readInt("Your move: "); 30        int computerMove = rg.nextInt(3); 31        int b = readInt("Computer move: ") + computerMove; 32 33        private int computer = 0; 34        private int human = 0; 35 36        private void roundWinner { 37            if (a == 1 &amp;&amp; b == 0) { 38                println("Human win"); 39            } 40        } 41    } 42 43    private void gameWinner { 44        if (human == 3) { 45            println("Human win"); 46        } else if (computer == 3) { 47            println("Computer win"); 48        } 49    } 50}</pre>

**Browse Answers**  
Grade: 26/30 points ✓ grade submitted

Rubric: Standard ▾

- For loop instead of a while loop (should go up to 3 wins by computer or user not 3 games)

User Inputting Move (3 points)

Perfect (0 points)  
 Minor Error (1 points)  
 Major Error (2 points)  
 Totally Wrong (3 points)

Computer Generating Move (4 points)

Perfect (0 points)  
 Minor Error (1 points)  
 Major Error (2 points)  
 Major Errors/No Attempt (4 points)

Determining the winner of a round (5 points)

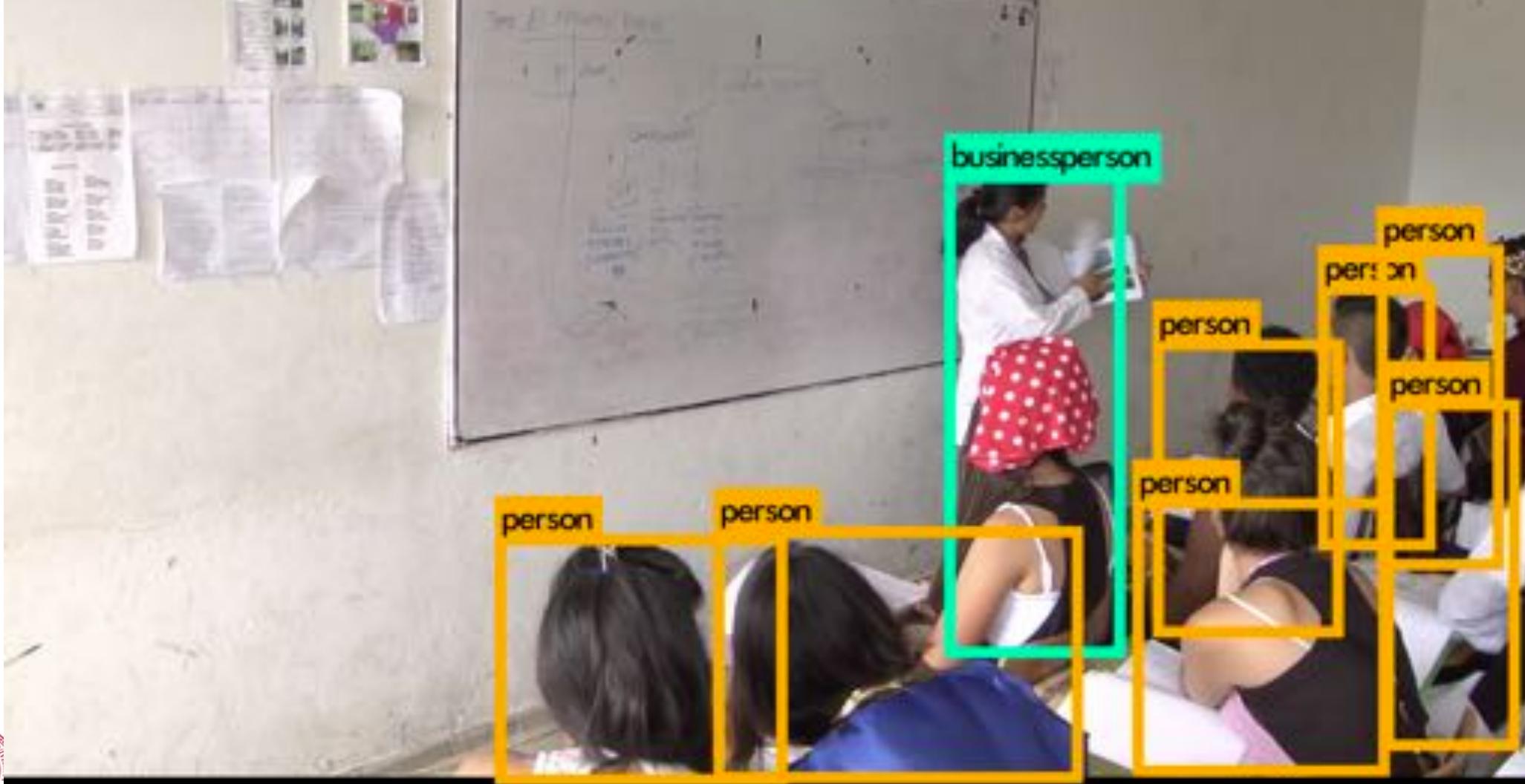
Perfect (0 points)  
 Minor Error (1 points)  
 Major Error (2 points)

< Previous   Submit Grade   Next >

Coming up  
**New problems**

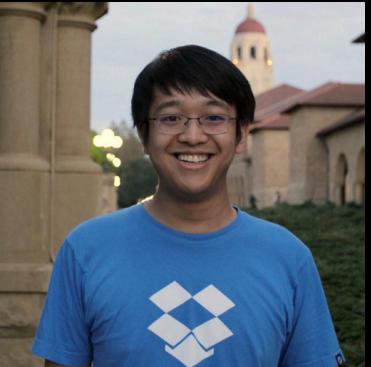
# Feedback for Teachers

**200,000** videos of teachers in Colombia, Chile and USA teaching



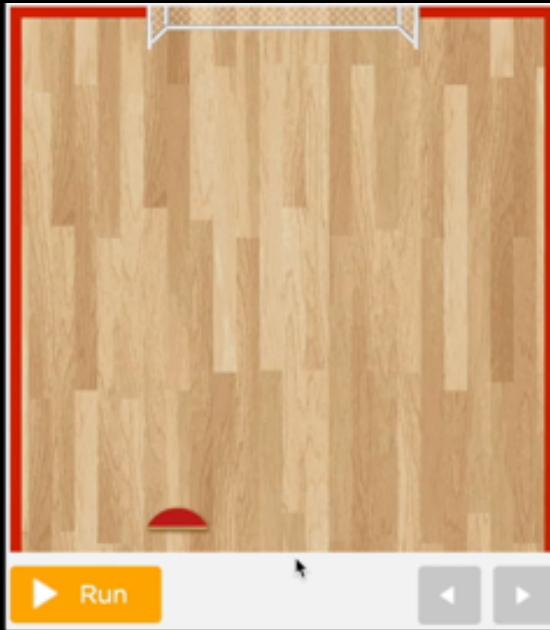
# Rotation: play to grade

Allen Nie



## Impact:

Immediately change  
what sort of assignments  
are auto-gradable



## Problem:

Grade the ~1M unique student  
implementations of this problem  
on code.org

## Input:

Teacher gives you one example of  
each mistake on their rubric and  
one example of invariances

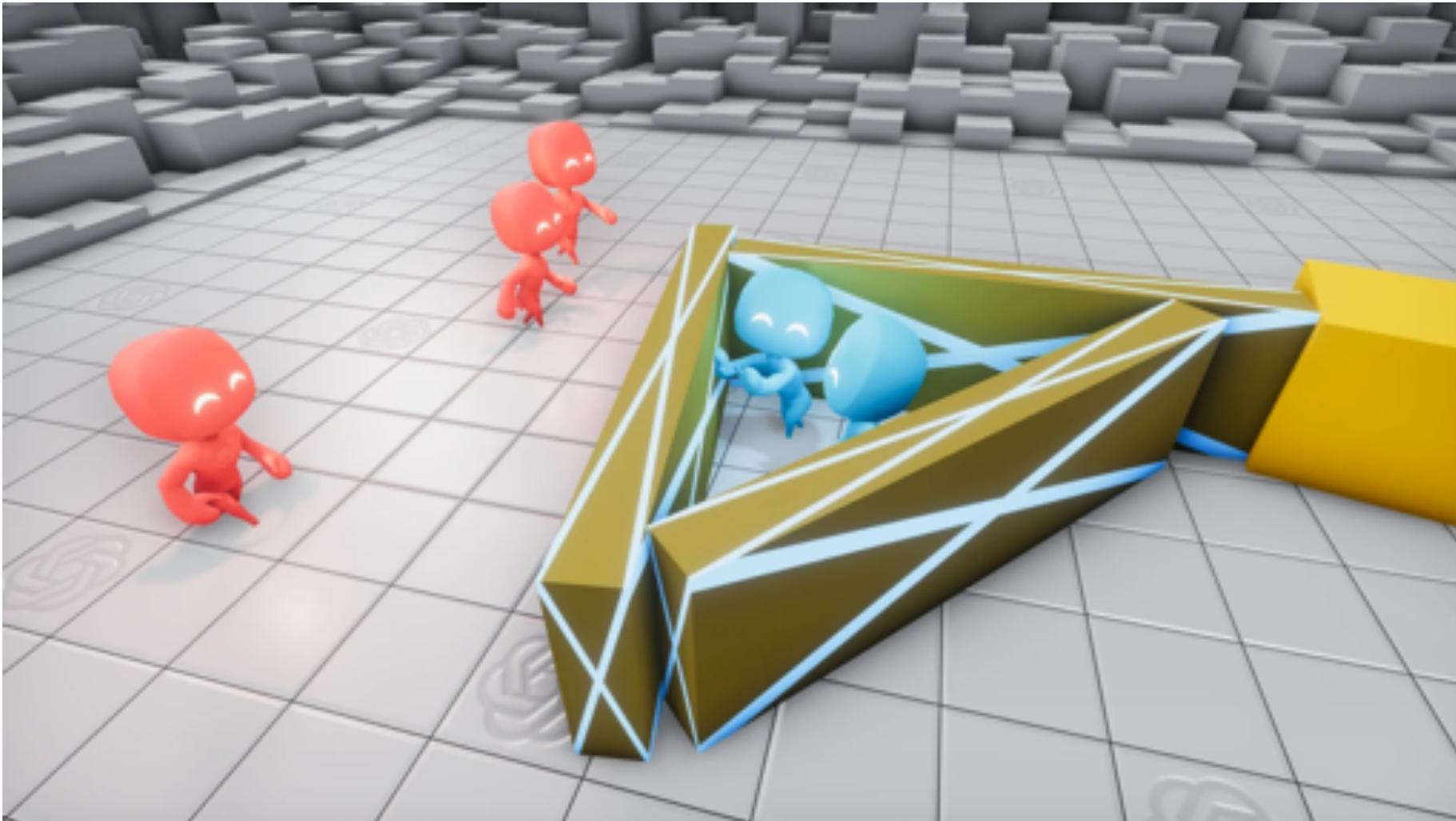
## Theory contribution:

First model to build deep RL for a  
classification task . *Instead of  
learning an environment you are  
learning to test an environment*



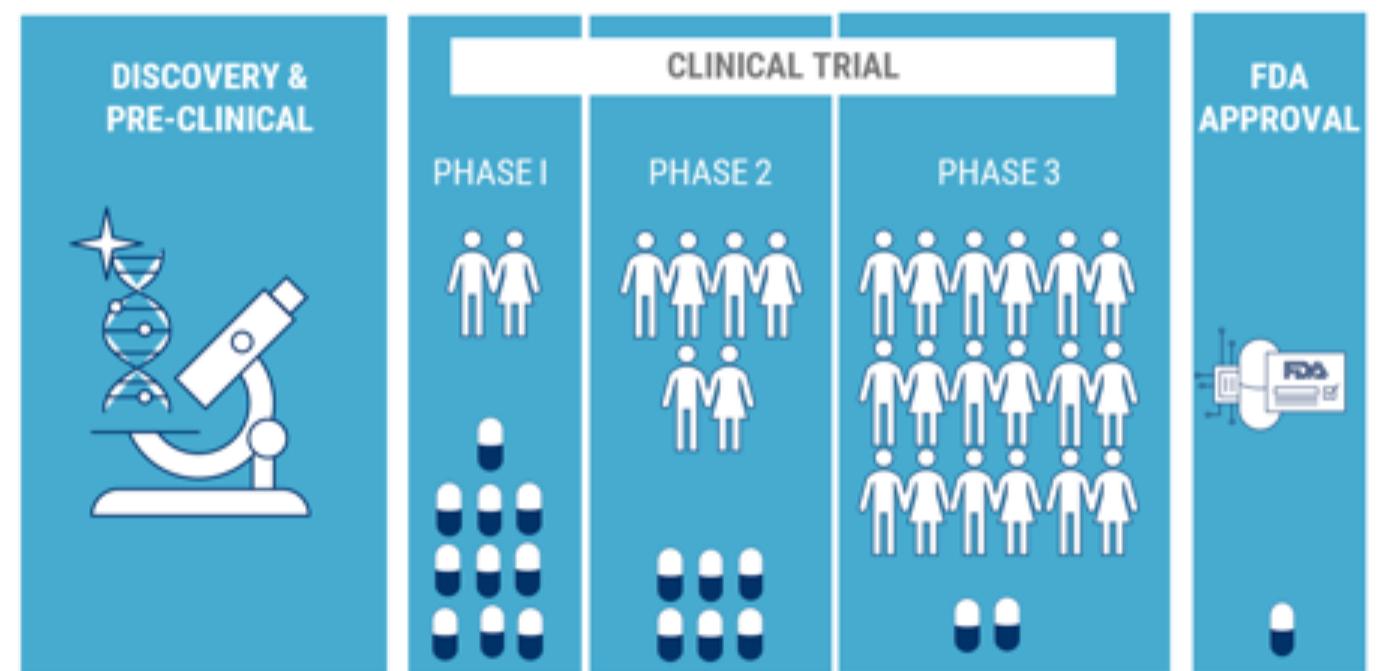
# Open Research Problems

## Teacher Gym



# More than education

 Bringing a drug to market is a drawn-out process



Source: cbinsights.com

 CBINSIGHTS



# Application -> Theory

Understand social science,  
especially with small data

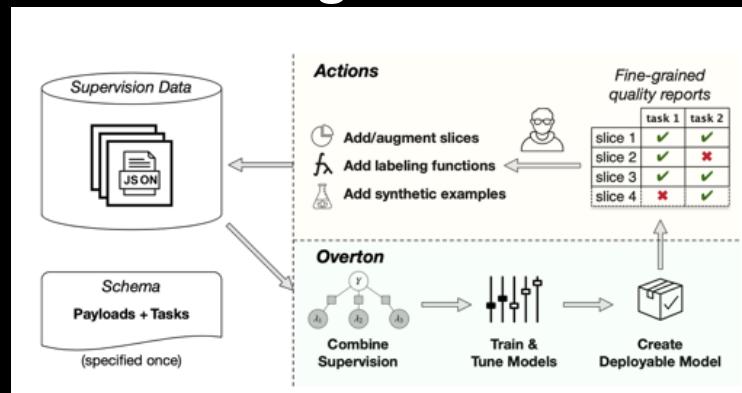
What are things that AI  
currently can't do?

What are things that AI  
currently can't do?

Design itself

Explain why it made the  
choices it did

Teach humans based on  
what it has learned



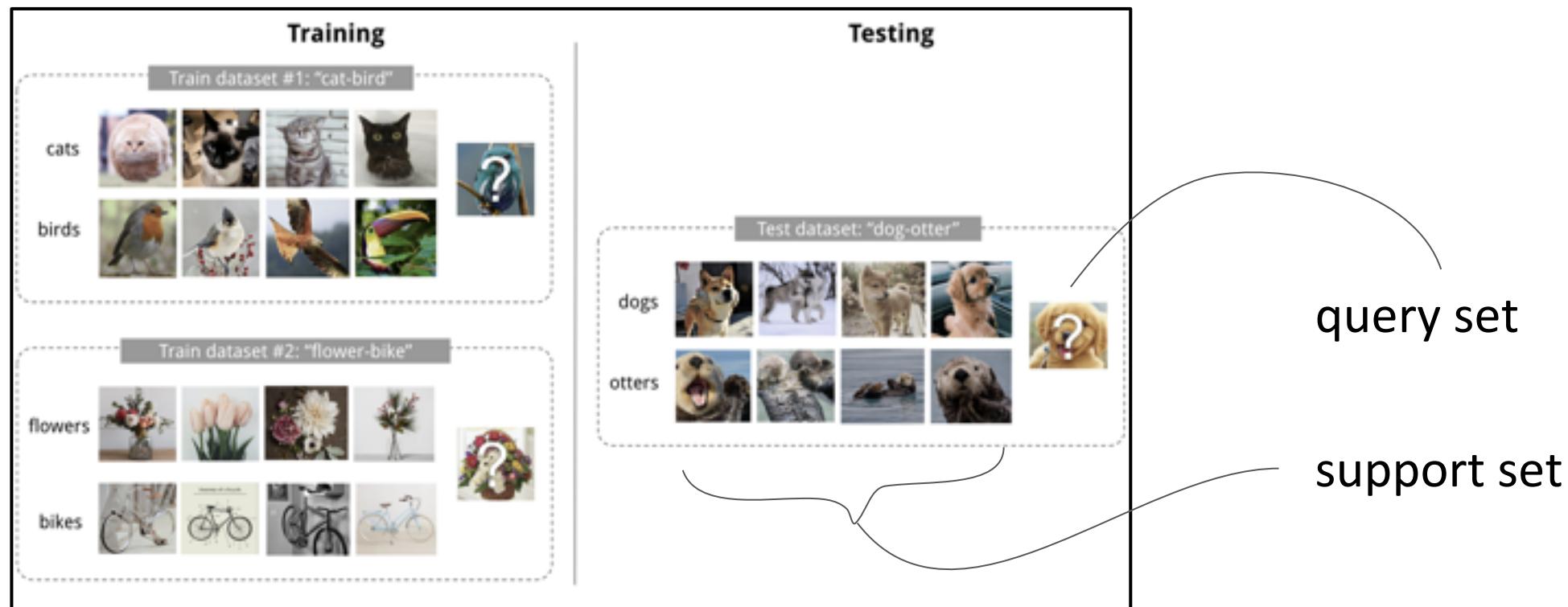
If time  
**Meta Learning**

Meta



Chelsea Finn

**Goal:** Divide **tasks** into meta-training and meta-test splits. Use the meta-training set to learn a model and evaluate performance on meta-test set.



## Meta Grading AI

Two different splits (“easy” and “hard”).

## Meta Grading AI

Two different splits (“easy” and “hard”).

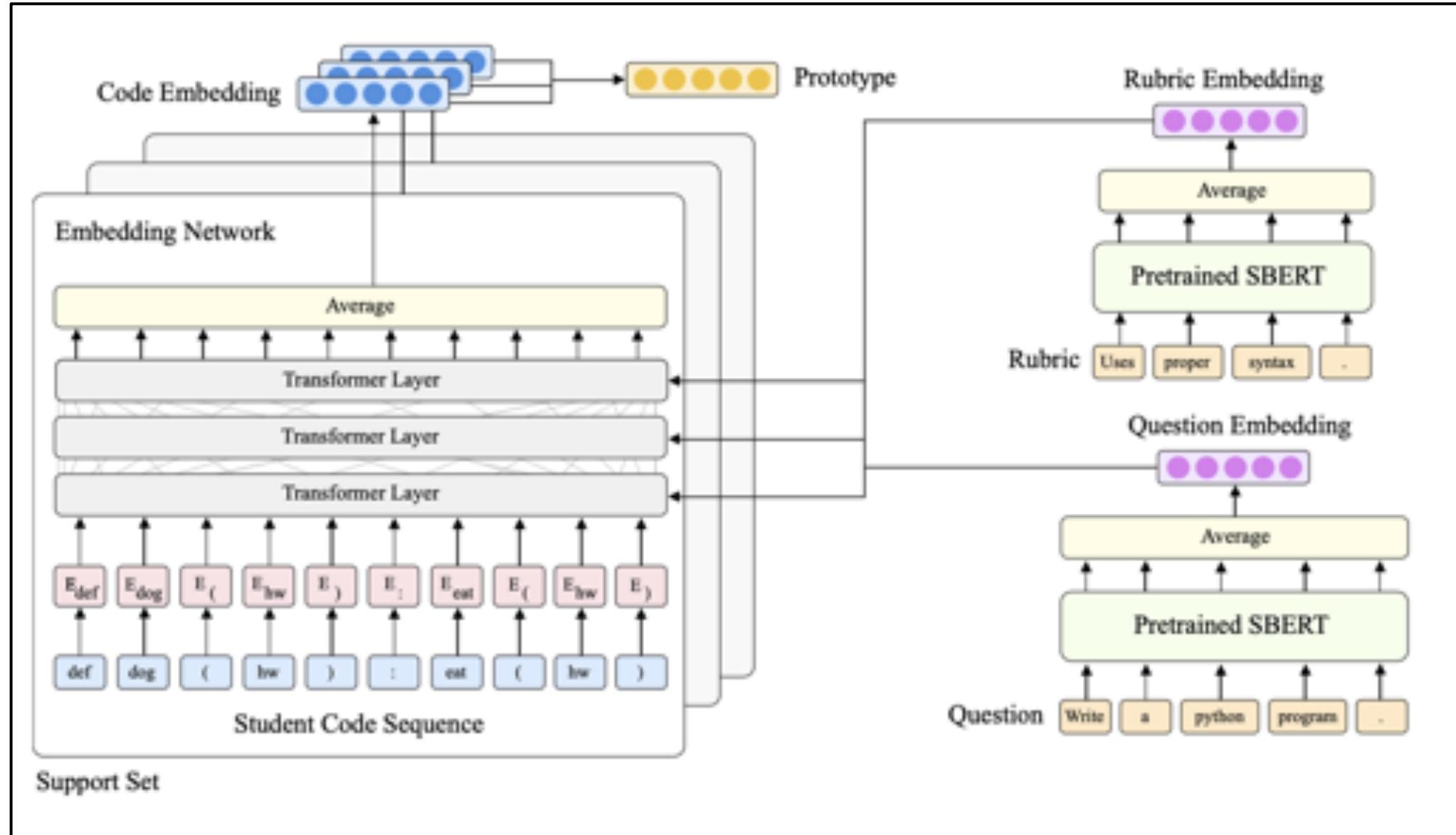
**Held-out rubric items** (“easy”): reserve 10% of rubric items for meta-test tasks.  
Same student programs can appear in both splits but for different items.

## Meta Grading AI

Two different splits (“easy” and “hard”).

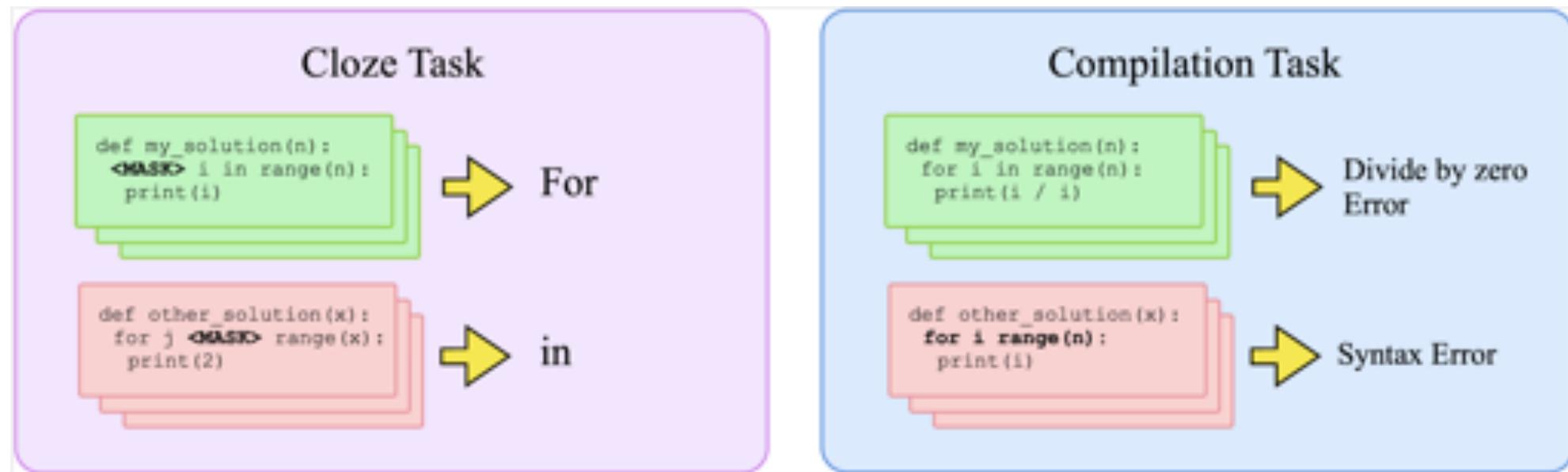
**Held-out rubric items** (“easy”): reserve 10% of rubric items for meta-test tasks.  
Same student programs can appear in both splits but for different items.

**Held-out exams** (“hard”): reserve 10% of exams for meta-test tasks. Truly unseen problems at test time.



## Trick #1: Task Augmentation

259 is not a lot of tasks. Meta-learning usually operates on 1000s of tasks. We apply the “data augmentation” idea to coding tasks!



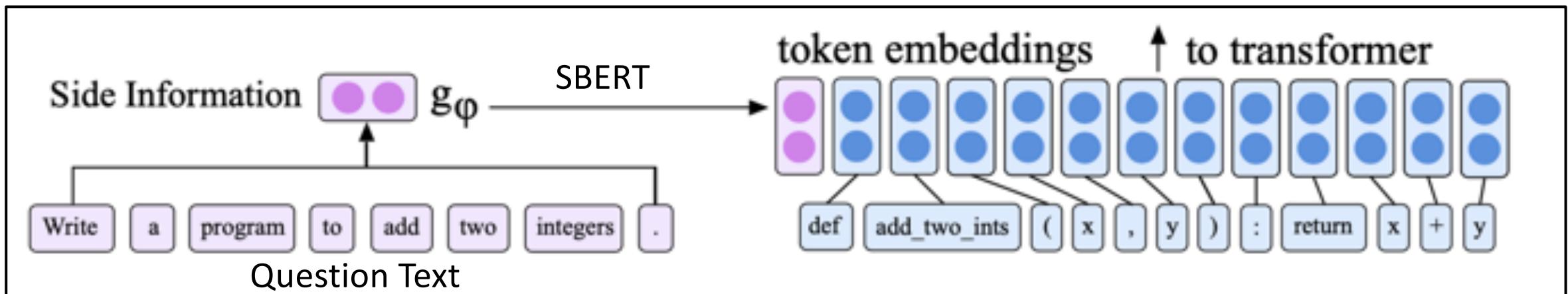
## Trick #2: Side Information

A task is only composed of 10 or 20 examples, leaving a lot of ambiguity. Suppose we have “side information”  $z = (z_1, z_2, \dots, z_T)$  about each task: **rubric option name** and **question text**. How do we add this side information into our embedding function  $f_\theta$ ?

## Trick #2: Side Information

A task is only composed of 10 or 20 examples, leaving a lot of ambiguity.

Suppose we have “side information”  $z = (z_1, z_2, \dots, z_T)$  about each task: **rubric** option name and **question text**. How do we add this side information into our embedding function  $f_\theta$ ?

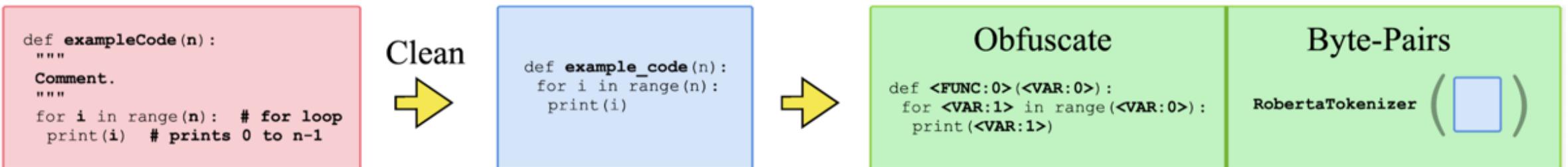


Prepend side information as a first token.

## Trick #3: Variable and Function Names

99% of our vocabulary is for variables and functions. This makes learning hard so we try two approaches.

1. Convert names to snakecase + use bytelpair encodings.
2. Allocate a fixed number of tokens that the model can use to represent variables and functions.

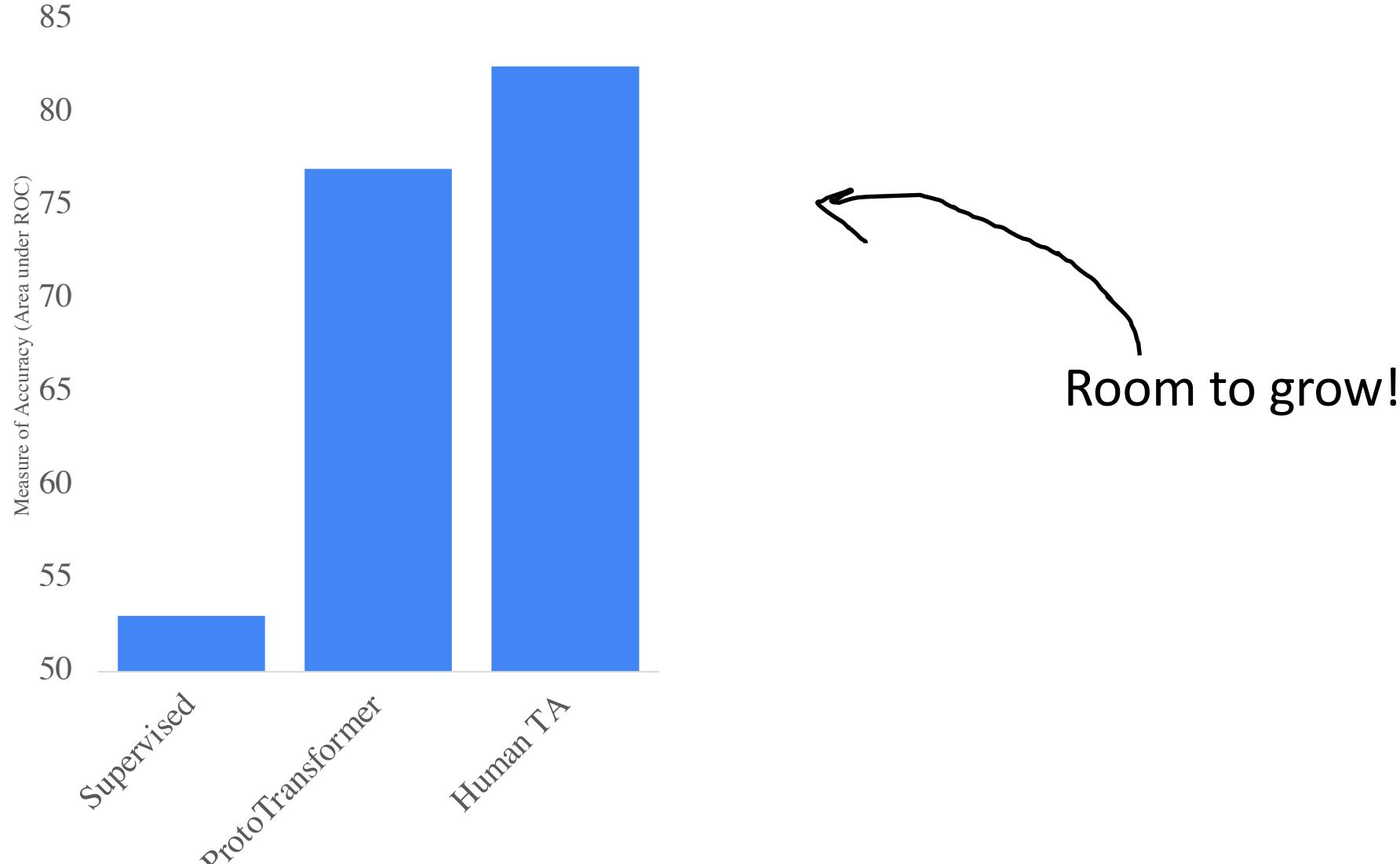


## Trick #4: Code Pre-training

Can we utilize large unlabeled datasets of code to help the model learn a good prior for code?

In practice, we initialize the embedding network from pretrain weights and finetune top M layers.

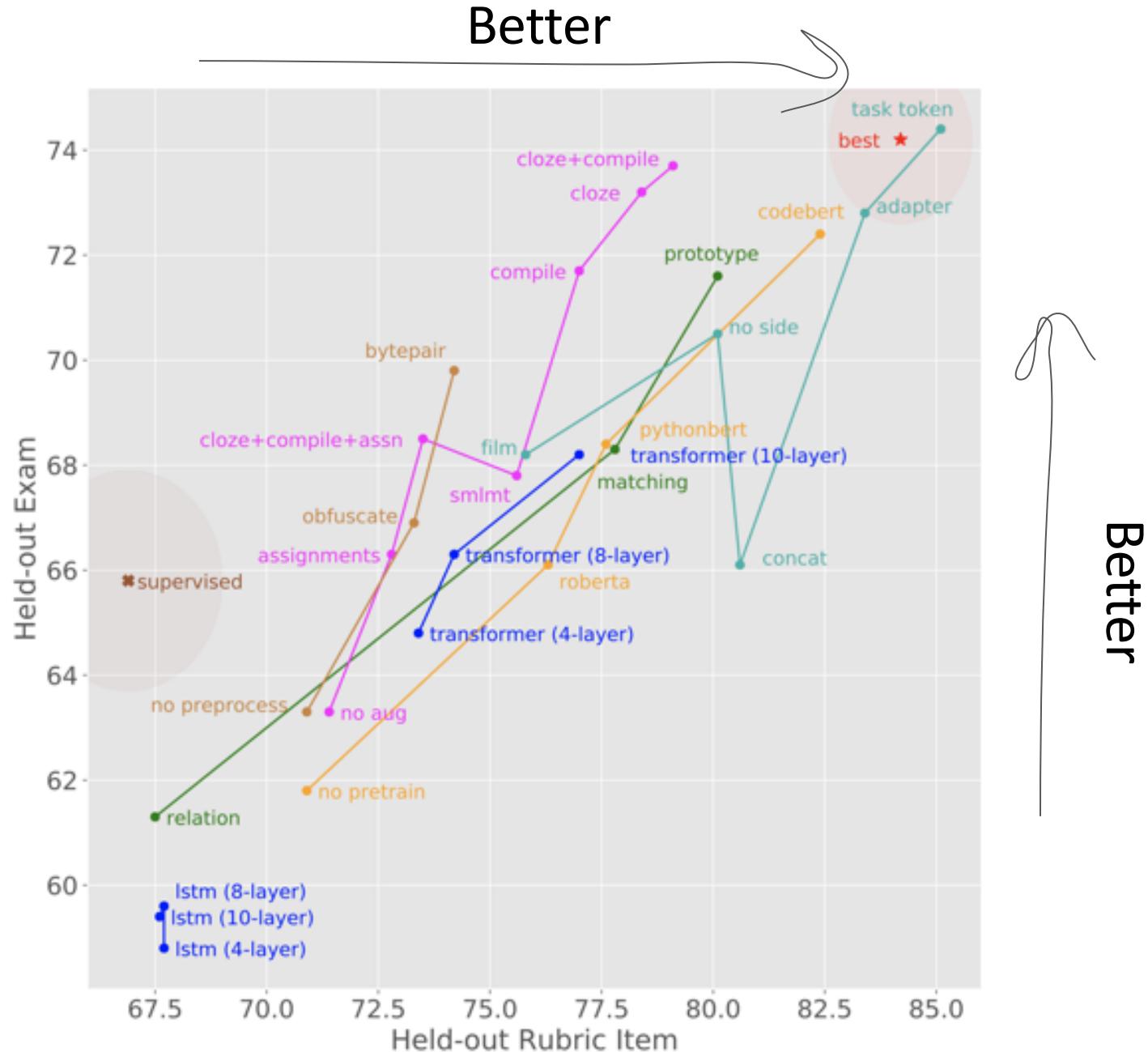
# Ability to Grade Unseen Exam (with 10 labels)



## Ablations

### Legend

- task aug
- naming
- architecture
- side info
- pretraining
- meta algo
- supervised
- best



# What's Next?

- **Do this at scale** (more data, more modalities, more diverse assignments). Collaboration with Gradescope?

Q1 [3pt] What is the integral of  $x^2$ ?

$x^2$

(STUDENT VIEW) ● GRADED

Midterm 1

TOTAL POINTS  
**6 / 8 pts**

QUESTION 1  
Calculus **4 / 6 pts**

1.1 Integral	<b>1 / 3 pts</b>
0 pts Correct	
✓ - 1 pt Missing 1/2	
✓ - 1 pt Missing Constant	
💬 See me after class tomorrow	
1.2 Derivative	<b>2 / 2 pts</b>

Download Submission History Request Regrade

Next Question ➔

# CS109

AI

Uncertainty Theory

Single Random  
Variables

Probabilistic  
Models

Counting

Probability Fundamentals

# Abundance of important problems

