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


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

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


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
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


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
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
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.

Smart Phone Access

Advanced Economies

Emerging Economies

- Smartphone
- Mobile
- No phone
Over 50 million learners

Unprecedented Data

Coursera

CODE

edX

Stanford University

Khan Academy

Udacity
US K-12 Students

= 500,000 learners
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
Clear
Societal Need

Grand Challenge in Education

Autonomously support education by better understanding students.

New Datasets of Learning

Online assignments

Deep Learning

AI Renaissance

Scale education
Feedback is Labor Intensive

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

Many domains of student work

Why did the original pilgrims come to America?
Chapter 0: Always start simple
First deep learning for education

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

Answer:
- Correct
- Incorrect

Khan AUC

- Baseline
- Old Gaurd
- Deep ML
Old Problem
Finding the x-intercept
Learns Concept Relationships

Finding the y-intercept
Learns Concept Relationships

Slope of a line
Graphing
Linear Equations
Learns Concept Relationships

Graphing Systems of Equations
Optimal Teaching

Average Predicted Probability vs. Exercise Index
Optimal Teaching

Maximize knowledge after 30 questions

Average Predicted Probability vs. Exercise Index

Exercise Index
Optimal Teaching

Maximize knowledge after 30 questions

Average Predicted Probability

Exercise Index

- Red: Blocking
- Blue: Mixing
Optimal Teaching

Maximize knowledge after 30 questions

Average Predicted Probability

Exercise Index

MDP-8
MDP-1
Blocking
Mixing
We truly would rather move beyond correct / incorrect
Some domains are very hard
Can you understand this code?

```java
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);
            }
        }
    }
}
```
Label student code

Traditional Deep Learning Doesn’t Work

Last Problem (P8)

<table>
<thead>
<tr>
<th>Feedback F1 Score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

- Old
- Gaurd
- Humans
Can we provide feedback by dynamic analysis?

- Starter code
- First attempt
- Final solution
Chapter 1: Better data source?
Each node is a unique partial solution.

Pink dots are students.

Each edge is what a teacher suggested.

Solution
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

18%
45%
12%
Desirable Path Algorithms

Poisson Common Path

\[ \gamma(s) = \text{First step in the most frequent path to the solution from } s, \text{ taken by average students. Assume poison process.} \]
Desirable Path Algorithms

Poisson Common Path

\[ \gamma(s) = \arg \min_{p \in \mathcal{Z}(s)} \sum_{x \in p} \frac{1}{\lambda x} \]

- Predicted next partial solution
- Paths to solution
- Partial solutions in the path
Learned Problem Solving Policy

Solution

“Backbone”
Only worked well for 6 line programs...
Chapter 2: Start to invent new algorithms...
// User defined method
private void run() {
    while (isClear()) {
        putBeeper();
        move();
    }
    putBeeper();
}

*Note: this was coded pre-tensor flow

It looks like you have a fencepost error!
Collect Triples

Precondition

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

Code

About 5 million triples per assignment
Collect Triples

Precondition

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

Code

Postcondition

About 5 million triples per assignment
A Code Phrase is a Mapping

All possible preconditions

All possible postconditions

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

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

Raw Precondition

Program Matrix

*coded pre-tensor flow
method step() {
    putBeeper();
    move();
}

*coded pre-tensor flow
Neural Network for Programs

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

*coded pre-tensor flow*
method step() {
    putBeeper();
    move();
}

Neural Network for Programs

Raw Precondition → Encoder → Multiply Program Matrix → Decoder → Raw Postcondition

*coded pre-tensor flow
method step() {
    putBeeper();
    move();
}

Neural Network for Programs

Raw Precondition \rightarrow \text{Encoder} \rightarrow \text{Multiply Program Matrix} \rightarrow \text{Decoder} \rightarrow \text{Raw Postcondition}

*coded pre-tensor flow

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

Neural Network for Programs

Raw Precondition → Encoder → Multiply Program Matrix → Decoder → Raw Postcondition

Autoencoding Loss → Prediction Loss

*coded pre-tensor flow
Does it work?
Traditional Deep Learning Doesn’t Work

Label student code

Last Problem (P8)

Feedback F1 Score

Old Gaurd

Humans
Inaccurate, Uninterpretable, and Data Hungry

Label student code

Last Problem (P8)

Feedback F1 Score

0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

Old Gaurd
Deep Learning
Humans

Piech et al, ICML 2014
Inaccurate, Uninterpretable, and Data Hungry

Last Problem (P8)

Feedback F1 Score

Old Gaurd
Deep Learning
Humans
Feedback F1 Score

Last Problem (P8)

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

\[ f(k) = \frac{1/k^s}{\sum_n (1/n^s)} \]
[Suspense]
Chapter 3: Back to the drawing board
Humans Don’t Need Much Data

Single training example:

Test set:
Fig. 1 People can learn rich concepts from limited data.


Copyright © 2015, American Association for the Advancement of Science
Fig. 2 Simple visual concepts for comparing human and machine learning.
Bayesian Program Learning
Fig. 4 Inferring motor programs from images.
Fig. 5 Generating new exemplars.
Fig. 6 Human and machine performance was compared on (A) one-shot classification and (B) four generative tasks.

Generative Understanding

Label student code

Last Problem (P8)

Feedback F1 Score

- Old Guard
- Deep Learning
- Humans

Struggle with double for loops
Confuses logic for deleting bricks
Imagine Students

• Struggle with double for loops
• Confuses logic for deleting bricks
Imagine Students

• Struggle with double for loops
• Confuses logic for deleting bricks

This is easy and exponential

\[ \Theta \sim \text{pythonSample} \]

\[ C \sim \text{pythonSample}|\Theta \]

\[ \Pi \sim \text{pythonSample}|C \]

This is hard and linear

\[ P(\Theta, C|\Pi) \]

A student’s “ability”

A student’s “choices”

The resulting code

Infer ability and choices from code
Bayesian Programming Language

ideaToText
Teachers Articulate N misconceptions

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

```python
# This python class is a RubricSampling Decision
# it generates programs that print the numbers 10 -> 1
class Countdown(Decision):
    def registerChoices(self):
        # these are the main strategies for printing out a
data
        self.addRubricChoices('loop-style', {
            'for': θ₁,
            'while': θ₂,
            'none': θ₃,
            'empty': θ₄
        })

    def processChoices(self):
        style = self.getChoice('loop-style')
        hasLoop = style != 'none' and style != 'empty'
        self.addLabel('rubric-hasLoop', hasLoop)

    def renderCode(self):
        style = self.getChoice('loop-style')
        if style == 'for': return '{ForSolt}'
        if style == 'while': return '{WhileSolt}'
        if style == 'none': return '{NoLoopSolt}'
        if style == 'empty': return ''
```
Generative Understanding

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

\[
p_G(x_{a_1}, \ldots x_{a_T} | y) = \prod_{t=1}^{T} p_G(x_{a_t} | y, x_{<a_t})
\]
Generative Understanding

Last Problem (P8)

- Old Guard
- Deep Learning
- Zero Shot Learning
- Humans

Feedback F1 Score

- Struggle with double for loops
- Confuses logic for deleting bricks

Outstanding Student paper award, AAAI 2019

Label student code

Struggle with double for loops
Confuses logic for deleting bricks
Not just for code

Results from early 2019
Many domains of student work

Why did the original colonists come to America?
So what?

What does this mean for me?!?
2,600 students
130,000 partial solutions
μ snapshots per student = 50
μ time per student = 2 hours

Step 1
Single brick

Step 2

Step 3

Step 4

Step 5

Step 6

Lisa Yan, Nick McKeown, Chris Piech. Understanding students from visual output. SIGCSE 2019
Students scoring in 99\textsuperscript{th} percentile on midterm exam

Students scoring in \leq 3\textsuperscript{rd} percentile on midterm exam

```
0
0.1
0.2
0.3
0.4

Fraction of commits

Other/Off-track (1, 8, 15, 16)
Stage 1: single row (2, 3, 4)
Stage 2: nested loop (5, 7)
Stage 3: adjusting nested offset (6, 9, 10, 11)
Stage 4: adding final details (12, 13, 14)
Error
```

t-SNE embedding of 130,000 partial solutions
One quarter, all students were shown their progress:

- Early correction of bad habits.
- Chance to teach the art of programming.
- Academic dishonesty becomes much harder.
Using assignment timing as pre-post

Item Response Theory based ability assessment

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

Predicted time Actual time \( p < 0.00001 \)

Assignments are taking less time

Students perform better than expected

\[ S_{i,j} = n \cdot \sigma(a_i - d_j) \]
Next trial

Code in Place 2021
But it's not perfect

Hit a ceiling with code > 20 lines
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:

```
0 1 2
```

```java
public class RockPaperScissors extends ConsoleProgram {
    // constants */
    private static final int ROCK = 0;
    private static final int PAPER = 1;
    private static final int SCISSORS = 2;
    private static final int N_WINS = 3;

    private RandomGenerator rg = new RandomGenerator();

    public void run() {
        introMessage();
        for (int i = 0; i < N_WINS; i++) {
            gameRound();
            gameWinner();
        }
        gameWinner();
    }

    private void introMessage() {
        println("Rock Paper Scissors!");
        println("(0) Rock");
        println("(1) Paper");
        println("(2) Scissors");
    }

    private void gameRound() {
        println(newGameMessage());
        int computerMove = rg.nextInt(3);
        println("Computer move: ");
    }

    private void gameWinner() {
        int winner = 0;
        if (computerMove == 0) {
            winner = 0;
            println("You win!");
        } else if (computerMove == 1) {
            winner = 1;
            println("Computer wins!");
        } else {
            winner = 2;
            println("Computer wins!");
        }
    }
}
```
Coming up

New problems
Feedback for Teachers

200,000 videos of teachers in Colombia, Chile and USA teaching
Input:
Teacher gives you one example of each mistake on their rubric and one example of invariances.

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

Impact:
Immediately change what sort of assignments are auto-gradable.

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

Rotation: play to grade
Open Research Problems

Teacher Gym
More than education

Bringing a drug to market is a drawn-out process

**Diagram:**
- ** Discovery & Pre-Clinical**
  - Phase I
  - Phase 2
  - Phase 3
- **Clinical Trial**
- **FDA Approval**

Source: cbinsights.com
Application -> Theory

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
- Understand social science, especially with small data
- What are things that AI currently can't do?
If time

Meta Learning
**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.
Two different splits ("easy" and "hard").
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.
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.
The diagram illustrates a system for code plagiarism detection. It involves a code embedding process followed by an embedding network that processes the student code sequence. The generated student code sequence is then used to generate a prototype. A rubric embedding is also created to assess the code against pre-trained SBERT models for questions like 'Uses proper syntax'. The overall system aims to ensure code quality and plagiarism detection using advanced text embedding techniques.
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” \( z = (z_1, z_2, ..., z_T) \) 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” \( z = (z_1, z_2, \ldots, z_T) \) 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 bytepair 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)

Room to grow!
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?
CS109

- AI
  - Uncertainty Theory
    - Single Random Variables
    - Probabilistic Models
      - Counting
      - Probability Fundamentals
Abundance of important problems