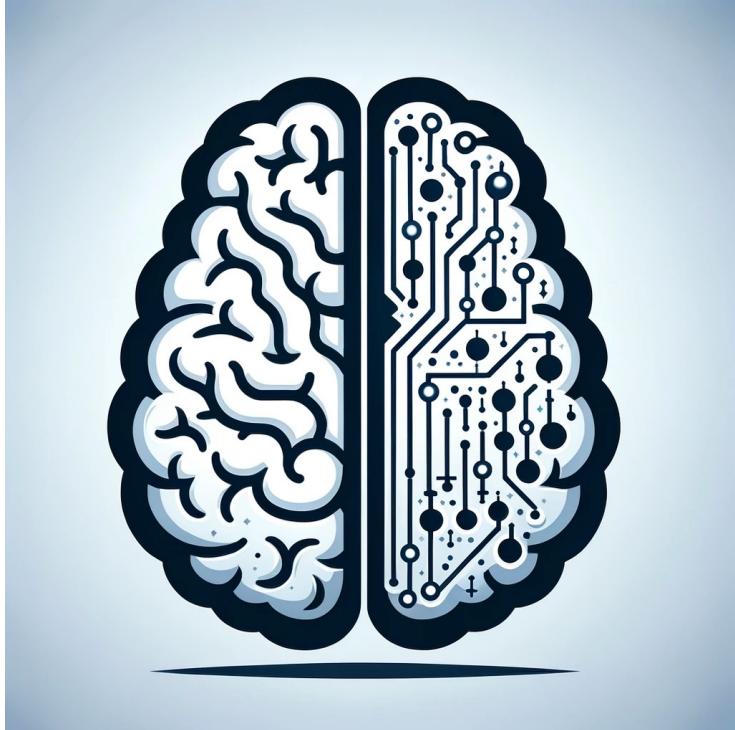


EN 601.473/601.673: Cognitive Artificial Intelligence (CogAI)



**Lecture 5:
Bayesian concept learning**

Tianmin Shu

Announcement

- New TA office hours: Wednesday, 8:30 – 9:30 am, Malone 216, starting from this week
- Spreadsheet for looking for project teammates

Readings

- Continue readings under week 2
- Go over the Julia tutorial (under week 3)

Outline

- Basic Bayesian cognition and Bayesian modeling
 - Theoretical foundations (modeling human inductive reasoning)
 - Flipping coins
 - Bayesian concept learning

Looking forward (beyond basic Bayes)

- Can we see these ideas at work in more natural cognitive function, not just toy problems and games?
 - How might differently structured hypothesis spaces, different likelihood functions or priors, be needed?
- Can we move from ‘weak rational analysis’ to ‘strong rational analysis’ in the priors, as with the likelihood?
 - “Weak”: behavior consistent with some reasonable prior.
 - “Strong”: behavior consistent with the “correct” prior given the structure of the world.
- Can we work with more flexible priors, not just restricted to a small subset of all logically possible concepts?
 - Would like to be able to learn any concept, even very complex ones, given enough data
 - E.g., Language-like hypothesis space?
- Can we describe formally how these hypothesis spaces and priors are generated by abstract knowledge or theories?
- Can we explain how people learn these rich priors?

Grounded language acquisition through the eyes and ears of a single child

WAI KEEN VONG  , WENTAO WANG  , A. EMIN ORHAN, AND BRENDEN M. LAKE  [Authors Info & Affiliations](#)

SCIENCE • 1 Feb 2024 • Vol 383, Issue 6682 • pp. 504-511 • DOI: 10.1126/science.ad1374



Time: 0:41

Utterance: You see this block the triangle



Time: 0:44

Utterance: It goes in



C



D



...



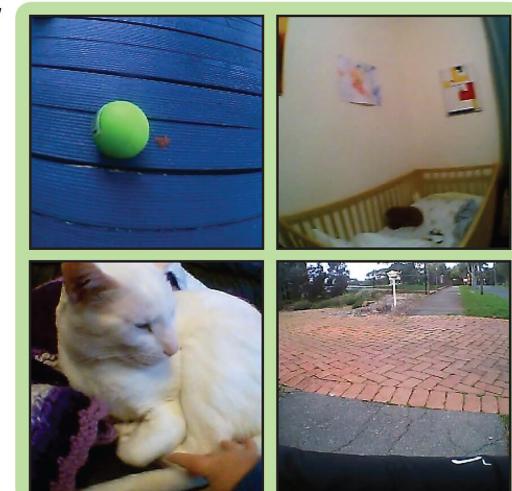
Time: 2:00

Utterance: You like the string

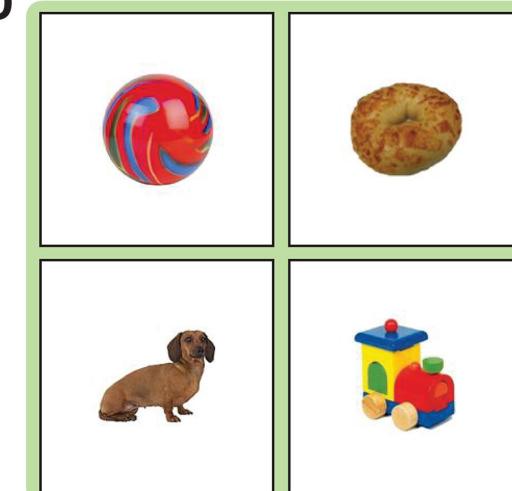


Time: 2:06

Utterance: You want the blocks too



D



Task: Which one is the ball?

Looking forward (beyond basic Bayes)

- Can we see these ideas at work in more natural cognitive function, not just toy problems and games?
 - How might differently structured hypothesis spaces, different likelihood functions or priors, be needed?
- Can we move from ‘weak rational analysis’ to ‘strong rational analysis’ in the priors, as with the likelihood?
 - “Weak”: behavior consistent with some reasonable prior.
 - “Strong”: behavior consistent with the “correct” prior given the structure of the world.
- Can we work with more flexible priors, not just restricted to a small subset of all logically possible concepts?
 - Would like to be able to learn any concept, even very complex ones, given enough data
 - E.g., Language-like hypothesis space?
- Can we describe formally how these hypothesis spaces and priors are generated by abstract knowledge or theories?
- Can we explain how people learn these rich priors?

Everyday prediction problems

- You stopped by a friend's apartment, and she has been watching a movie for 15 minutes. What would you predict for the length of the movie in total?
- You stopped by a friend's apartment, and she has been watching a movie for 75 minutes. What would you predict for the length of the movie in total?
- A movie has grossed 15 million dollars at the box office, but you don't know how long it's been running. How much will it gross in total?
- You encounter a phenomenon or event with an unknown extent or duration, t_{total} , at a random time or value of $t_{past} < t_{total}$. What is the total extent or duration t_{total} ?

Bayesian modeling

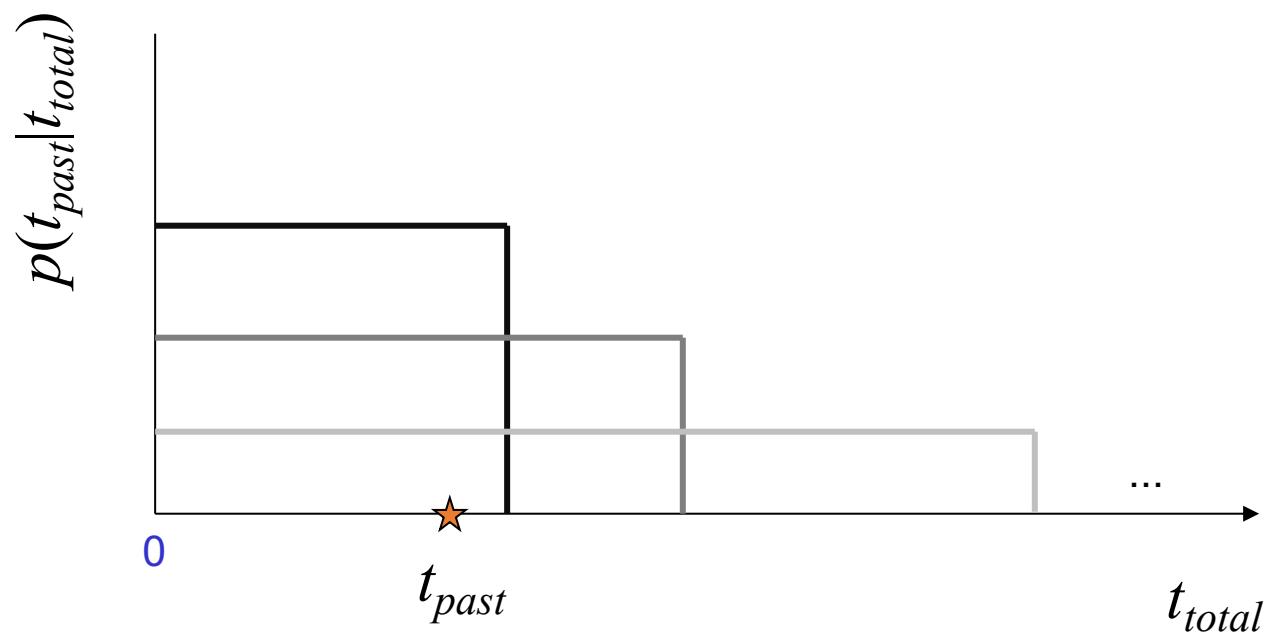
$$p(t_{total}|t) \propto p(t|t_{total})p(t_{total})$$

$$p(t|t_{total}) = \frac{1}{t_{total}}$$

Likelihood

$$p(t|t_{total}) = \frac{1}{t_{total}}$$

Assume uniformly sample t from the range $(0, t_{total})$



Bayesian modeling

$$p(t_{total}|t) \propto p(t|t_{total})p(t_{total})$$

$$p(t|t_{total}) = \frac{1}{t_{total}}$$

$$p(t_{total}|t) \propto \frac{p(t_{total})}{t_{total}}$$

Form of $p(t_{total})$?

Uninformative prior

- Not matter what domains (movie gross, movie runtimes, etc.), a single uninformative prior:

$$p(t_{total}) \propto 1/t_{total}$$

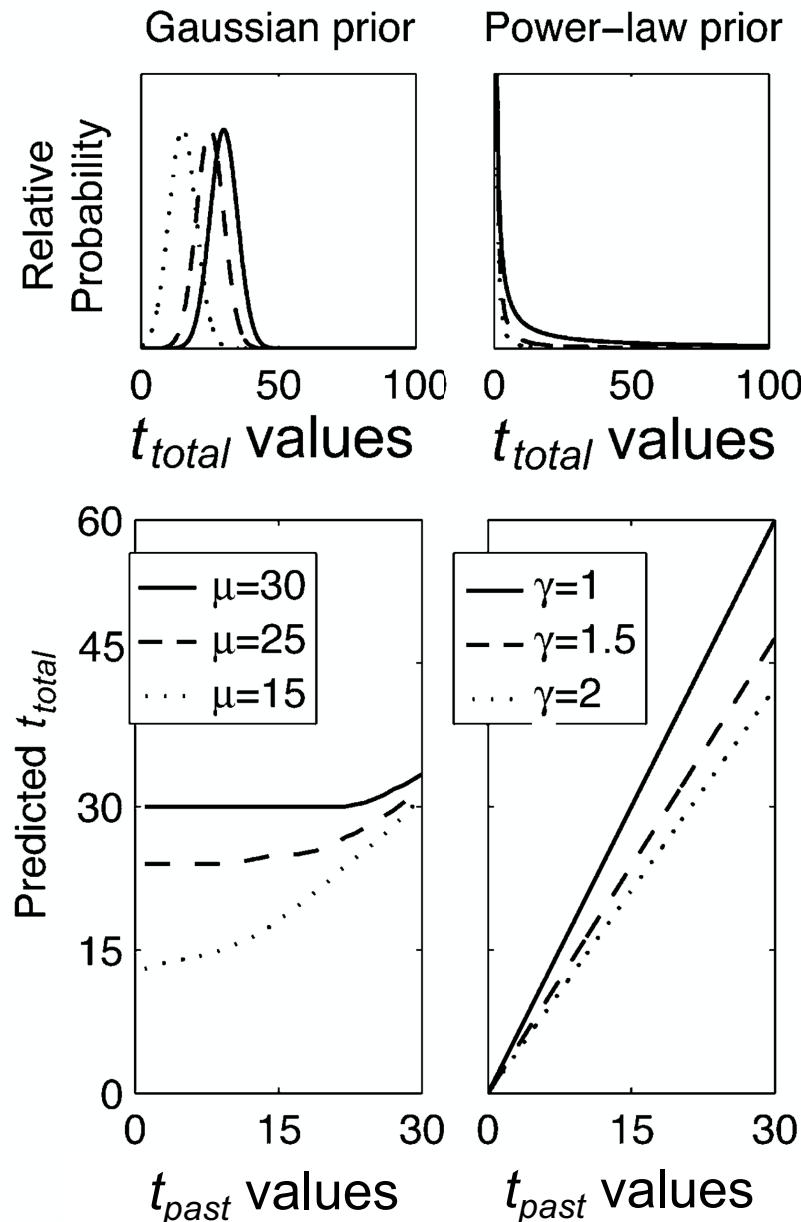
nature Implications of the Copernican principle for our future prospects

[J. Richard Gott III](#)

[Nature](#) 363, 315–319 (1993) | [Cite this article](#)

Making only the assumption that you are a random intelligent observer, limits for the total longevity of our species of 0.2 million to 8 million years can be derived at the 95% confidence level. Further consideration indicates that we are unlikely to colonize the Galaxy, and that we are likely to have a higher population than the median for intelligent species.

Different priors have qualitatively different predictions



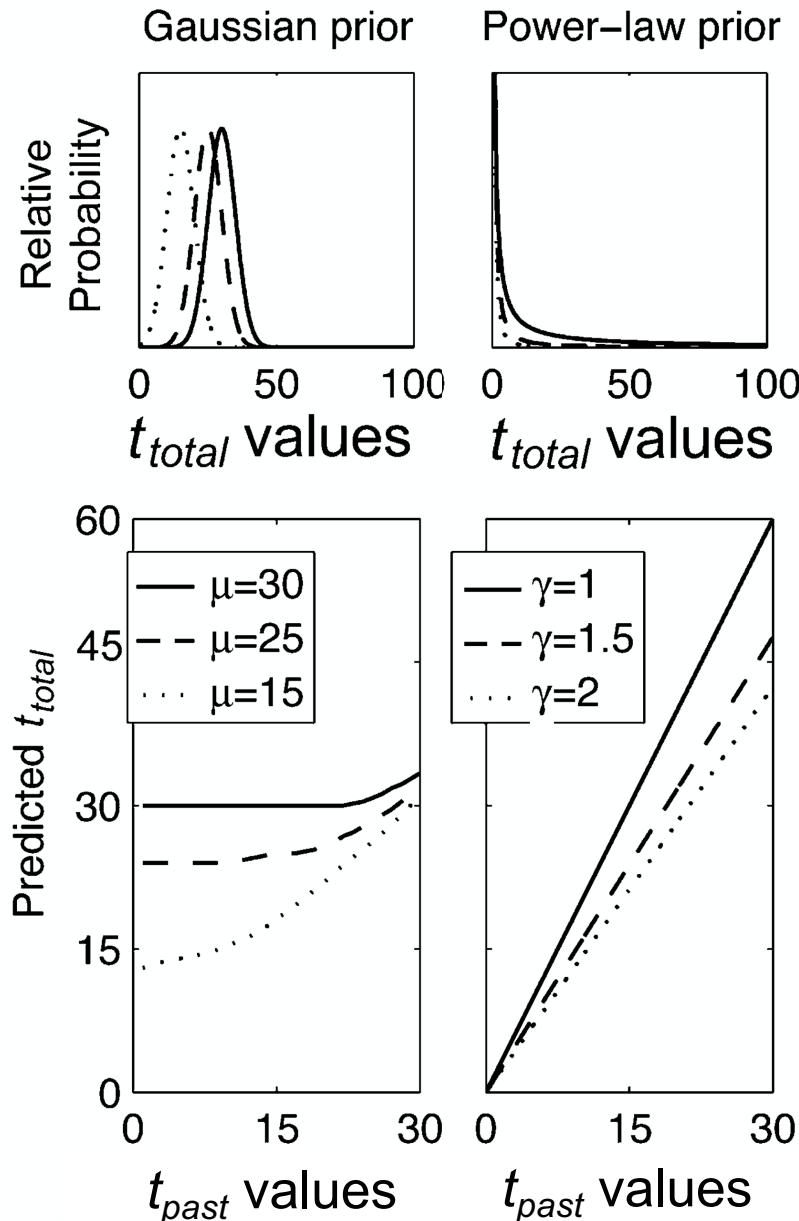
Gaussian prior:

$$P(t_{total}) \propto \exp\left(-\frac{1}{2\sigma^2}(t_{total} - \mu)^2\right)$$

Power-law prior:

$$P(t_{total}) \propto t_{total}^{-\gamma}$$

Different priors are appropriate in different domains



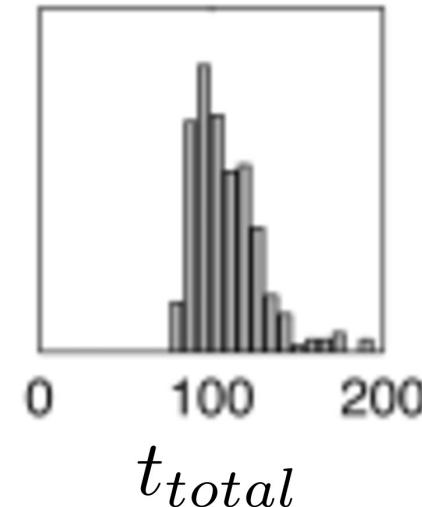
Gaussian prior:

$$P(t_{total}) \propto \exp\left(-\frac{1}{2\sigma^2}(t_{total} - \mu)^2\right)$$

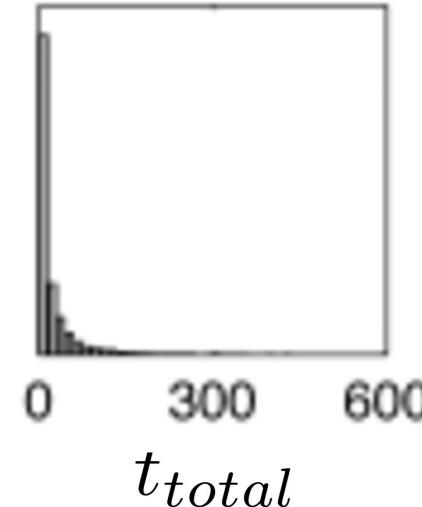
Power-law prior:

$$P(t_{total}) \propto t_{total}^{-\gamma}$$

**Movie runtimes
(Gaussian)**



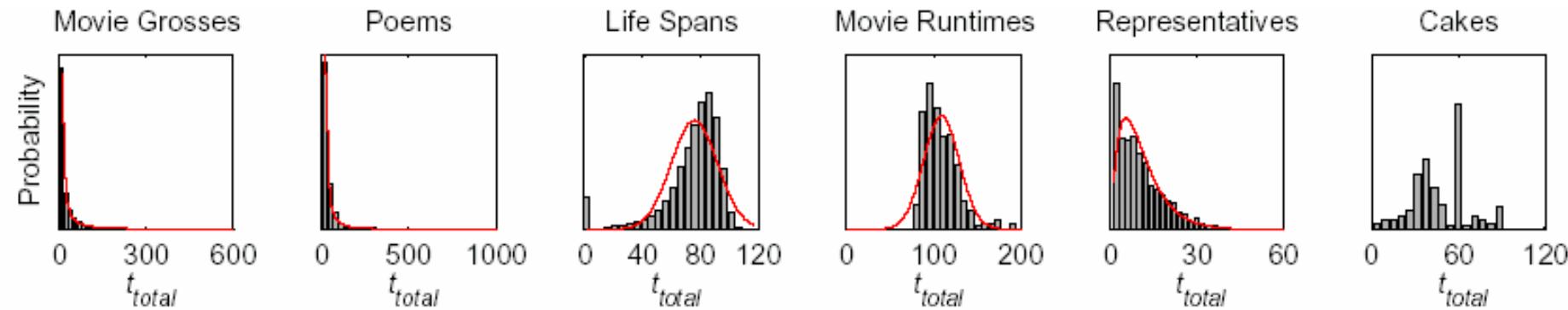
**Movie grosses
(Power-law)**



Evaluating human predictions in different domains

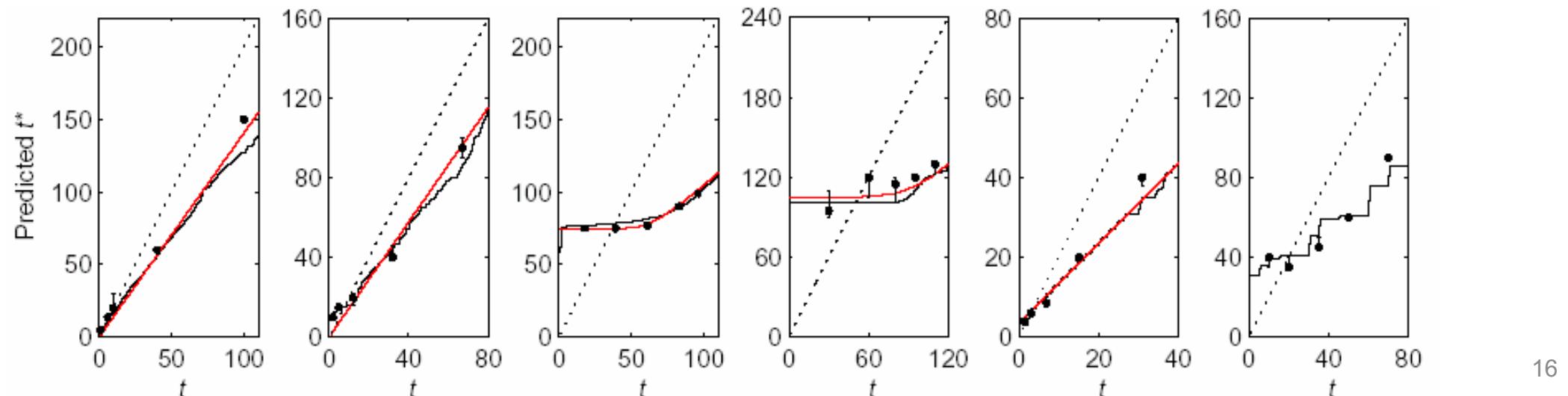
- **Movie gross:** You read about a movie that has made \$60 million to date. How much money will it make in total?
- **Poem lengths:** Your friend quotes to you from line 17 of his favorite poem. How long is the poem?
- **Life spans:** You meet someone who is 78 years old. How long will they live?
- **Movie runtimes:** You stopped by a friend's apartment, and she has been watching a movie for 55 minutes. What would you predict for the length of the movie in total?
- **Terms of U.S. representatives:** You meet a US congressman who has served for 11 years. How long will he serve in total?
- **Baking times for cakes:** You see that something has been baking in the oven for 34 minutes. How long until it's ready?
- Use values of t_{past} for each domain, creating 5 events in each domain
- People predict t_{total} for each event

Priors $p(t_{total})$ based on empirically measured durations or magnitudes for many real-world events in each class:



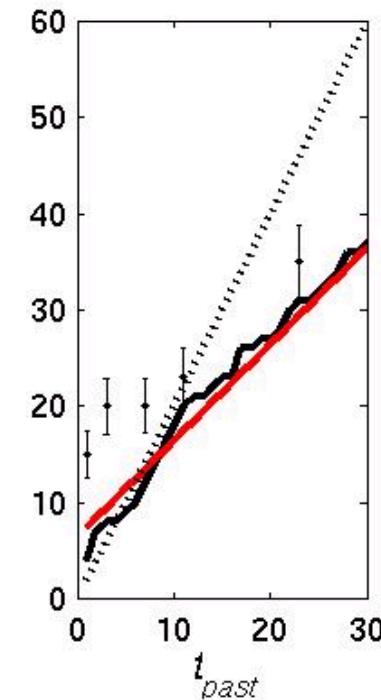
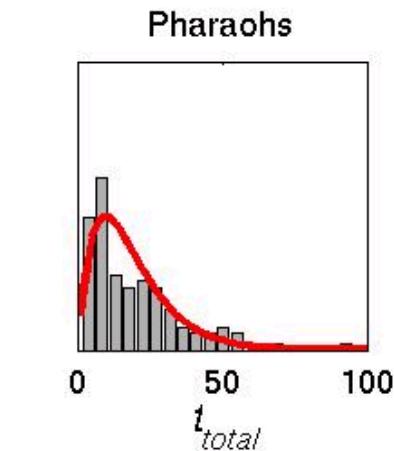
Median human judgments of the total duration or magnitude t_{total} of events in each class, given one random observation at a duration or magnitude t , versus Bayesian predictions (median of $p(t_{total}|t)$).

Dot: human
Solid line: emp. priors
Dashed line: uninform. prior



A special case

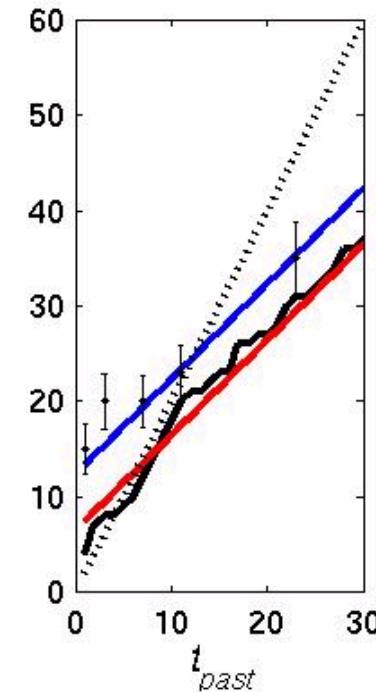
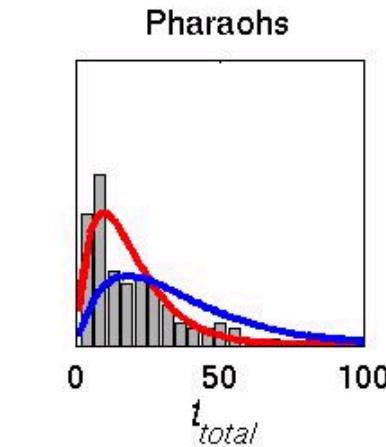
- Reigns of pharaohs: You learned that in ancient Egypt, there was a great flood in the 11th year of a pharaoh's reign. How long did he reign?



A special case

- Reigns of pharaohs: You learned that in ancient Egypt, there was a great flood in the 11th year of a pharaoh's reign. How long did he reign?

- Ask 35 undergraduate students:
How long did the typical pharaoh
reign in ancient Egypt?
- Recalibrate prior based on their
responses



Is cognition rational?

- Strong version of Bayesian cognition:
 - Is cognition well described by an **optimal** (Bayesian) model of inference, using the “right” priors and likelihoods?
 - Is this possible?
 - At least in some cases, perhaps yes.
 - Priors vary with the qualitative form (power-law vs. exponential) + correct parameter values
 - People seem to accurately absorb the statistics of their environment for everyday quantitates
 - In ML/AI: learning priors from prior experience
 - In the absence of concrete experience (e.g., Pharaoh), priors may be generated by qualitative background knowledge
 - In ML/AI: query a knowledge base / language model

Looking forward

- Can we see these ideas at work in more natural cognitive function, not just toy problems and games?
 - How might differently structured hypothesis spaces, different likelihood functions or priors, be needed?
- Can we move from ‘weak rational analysis’ to ‘strong rational analysis’ in the priors, as with the likelihood?
 - “Weak”: behavior consistent with some reasonable prior.
 - “Strong”: behavior consistent with the “correct” prior given the structure of the world.
- Can we work with more flexible priors, not just restricted to a small subset of all logically possible concepts?
 - Would like to be able to learn any concept, even very complex ones, given enough data
 - E.g., **Language-like hypothesis space?**
- Can we describe formally how these hypothesis spaces and priors are generated by abstract knowledge or theories?
- Can we explain how people learn these rich priors and enable AI to do so?

Let's revisit the number game

- A question from one of you:
- Can we compose more complex hypotheses?

The need for more flexible priors

- Suppose you see these examples in the number game:
 - 60, 22, 80, 10, 25, 30, 24, 90, 27, 70, 26, 29
 - Multiples of 10 and 20 – 30

The need for more flexible priors

- Suppose you see these examples in the number game:
 - 60, 22, 80, 10, 25, 30, 24, 90, 27, 70, 26, 29
 - 60, 80, 10, 30, 40, 20, 90, 80, 60, 40, 10, 20, 80, 30, 90, 60, 40, 30, 60, 80, 20, 90, 10, 30, 40, 90, 10, 60, 20, 80, 30
 - Multiples of 10
 - Multiples of 10 except 50 and 70

The need for more flexible priors

- Suppose you see these examples in the number game:
 - 60, 22, 80, 10, 25, 30, 24, 90, 27, 70, 26, 29
 - 60, 80, 10, 30, 40, 20, 90, 80, 60, 40, 10, 20, 80, 30, 90, 60, 40, 30, 60, 80, 20, 90, 10, 30, 40, 90, 10, 60, 20, 80, 30
 - 60, 80, 10, 30, 52
 - Multiples of 10, plus 52

The need for more flexible priors

- Suppose you see these examples in the number game:
 - 60, 22, 80, 10, 25, 30, 24, 90, 27, 70, 26, 29
 - 60, 80, 10, 30, 40, 20, 90, 80, 60, 40, 10, 20, 80, 30, 90, 60, 40, 30, 60, 80, 20, 90, 10, 30, 40, 90, 10, 60, 20, 80, 30
 - 60, 80, 10, 30, 52
 - 60, 80, 10, 30, 55
 - Multiples of 10, plus 55?
 - Multiples of 5?

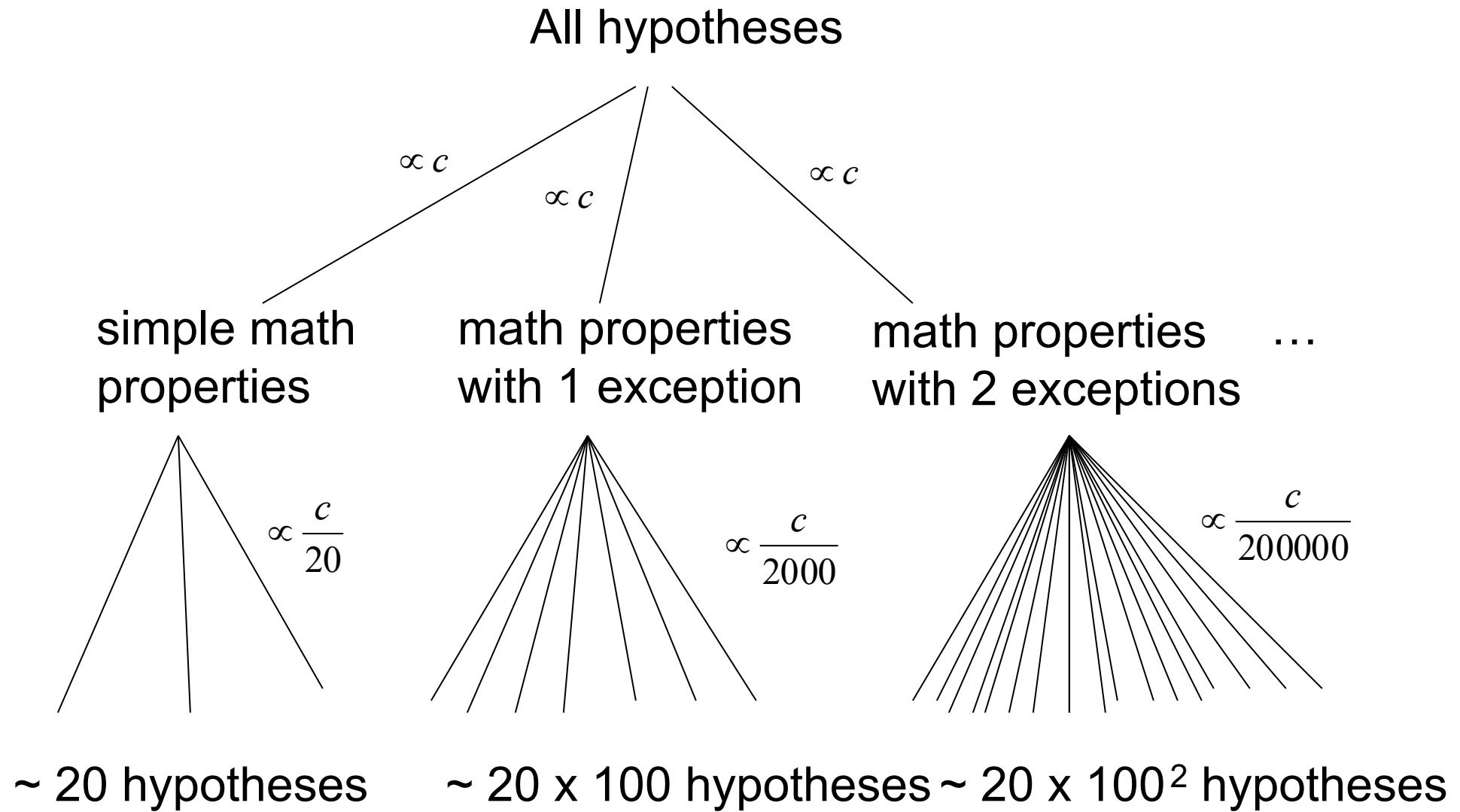
The need for more flexible priors

- Suppose you see these examples in the number game:
 - 60, 22, 80, 10, 25, 30, 24, 90, 27, 70, 26, 29
 - 60, 80, 10, 30, 40, 20, 90, 80, 60, 40, 10, 20, 80, 30, 90, 60, 40, 30, 60, 80, 20, 90, 10, 30, 40, 90, 10, 60, 20, 80, 30
 - 60, 80, 10, 30, 52
 - 60, 80, 10, 30, 55
 - 60, 80, 10, 30, 55, 40, 20, 80, 50, 10
 - Multiples of 10, plus 55
 - Multiples of 5

Constructing more flexible priors: A language for thought?

- Start with a base set of regularities R and combination operators C .
Hypothesis space = closure of R under C .
 - $C = \{\text{and}, \text{or}\}$: H = unions and intersections of regularities in R (e.g., “multiples of 10 between 30 and 70”).
- Start with a base set of regularities R and allow singleton (+ or -) exceptions.
 - e.g., “multiples of 10 except 50”, “multiples of 10 except 50 and 70”, “multiples of 10 and also 52”.
- Defining a prior:
 - The Bayesian Occam’s Razor:
 - Model classes defined by number of combinations.
 - More combinations → more hypotheses → lower prior

Prior



Another type of language for thought: programs

- lambda function (functional programming):

(**Lambda** (*arg*) (+ *arg* 1))

- Functional composition: Let *x* be a stream of natural numbers {1, 2, 3, ...}

A DSL (domain-specific language) Programs as hypotheses

(* 3 *x*): multiples of 3

(<= 30 (>= 20 *x*)): numbers between 20 and 30

(* 2 *x*): even numbers

(<= 15 (>= 15 *x*)): numbers between 10 and 15

(+ 1 (* 2 *x*)): odd numbers

(<= 30 (>= 20 (* 2 *x*))): even numbers between 20 and 30

(power_of 2 *x*): powers of 2

(>= 20 (* 2 *x*)): even numbers greater than 20

(power_to 2 *x*): square numbers

(power-of 2 (+ 1 (* 2 *x*))): odd-numbered powers of 2

(<= 10 *x*): numbers less than 10

(<= 50 (* 10 *x*)): multiples of 10 up to 100

(= 80 *x*): exactly 80

(+ 3 (* 4 *x*)): multiples of 4, shifted by 3 (3, 7, 11, 15, ...)

(!= 70 *x*): numbers except 70

(!= 50 (!= 70 (* 10 *x*))): multiples of 10 except 50 and 70

Run the program to get all valid numbers → likelihood based on size principle

*Could this approach be extended to all the learnable concepts?
If not, what else is needed? A possible project topic.*

Program induction for concept learning

- Given a domain with multiple concepts, come up with a DSL and write programs to interpret different concepts (concept = program)

RESEARCH ARTICLES

COGNITIVE SCIENCE

Human-level concept learning through probabilistic program induction

Brenden M. Lake,^{1*} Ruslan Salakhutdinov,² Joshua B. Tenenbaum³

People learning new concepts can often generalize successfully from just a single example, yet machine learning algorithms typically require tens or hundreds of examples to perform with similar accuracy. People can also use learned concepts in richer ways than conventional algorithms—for action, imagination, and explanation. We present a computational model that captures these human learning abilities for a large class of simple visual concepts: handwritten characters from the world’s alphabets. The model represents concepts as simple programs that best explain observed examples under a Bayesian criterion. On a challenging one-shot classification task, the model achieves human-level performance while outperforming recent deep learning approaches. We also present several “visual Turing tests” probing the model’s creative generalization abilities, which in many cases are indistinguishable from human behavior.

Science

1-shot handwritten character recognition

ଏ	ର୍ବ	ଲୁ	ଦୟ	ଜ୍ଞ
କେ	ନୁ	ଗୁ	ଚୁ	ଦ୍ୟୁ
ଖୁ	ତୁ	ଛୁ	ତେ	ଦ୍ୱା
ନେ	ଯୁ	ଲୁ	ର୍ଯ୍ୟ	ଭୁ

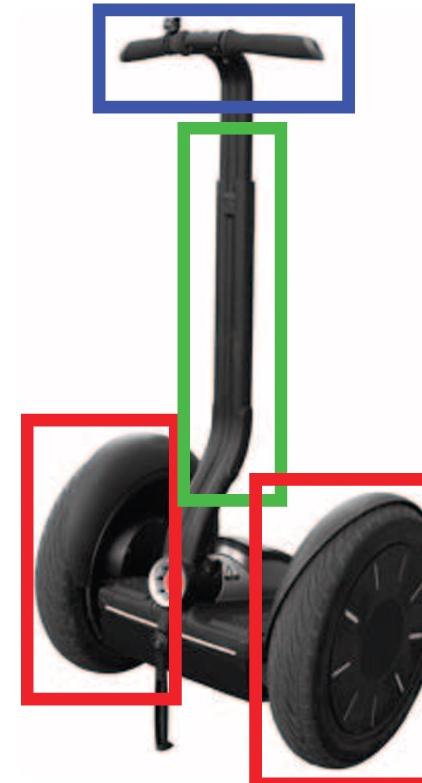
Program induction for concept learning

- Given a domain with multiple concepts, come up with a DSL and write programs to interpret different concepts (concept = program)

Segway



Human constraint / inductive bias / prior:
Parsing an object into **parts** and **relations**

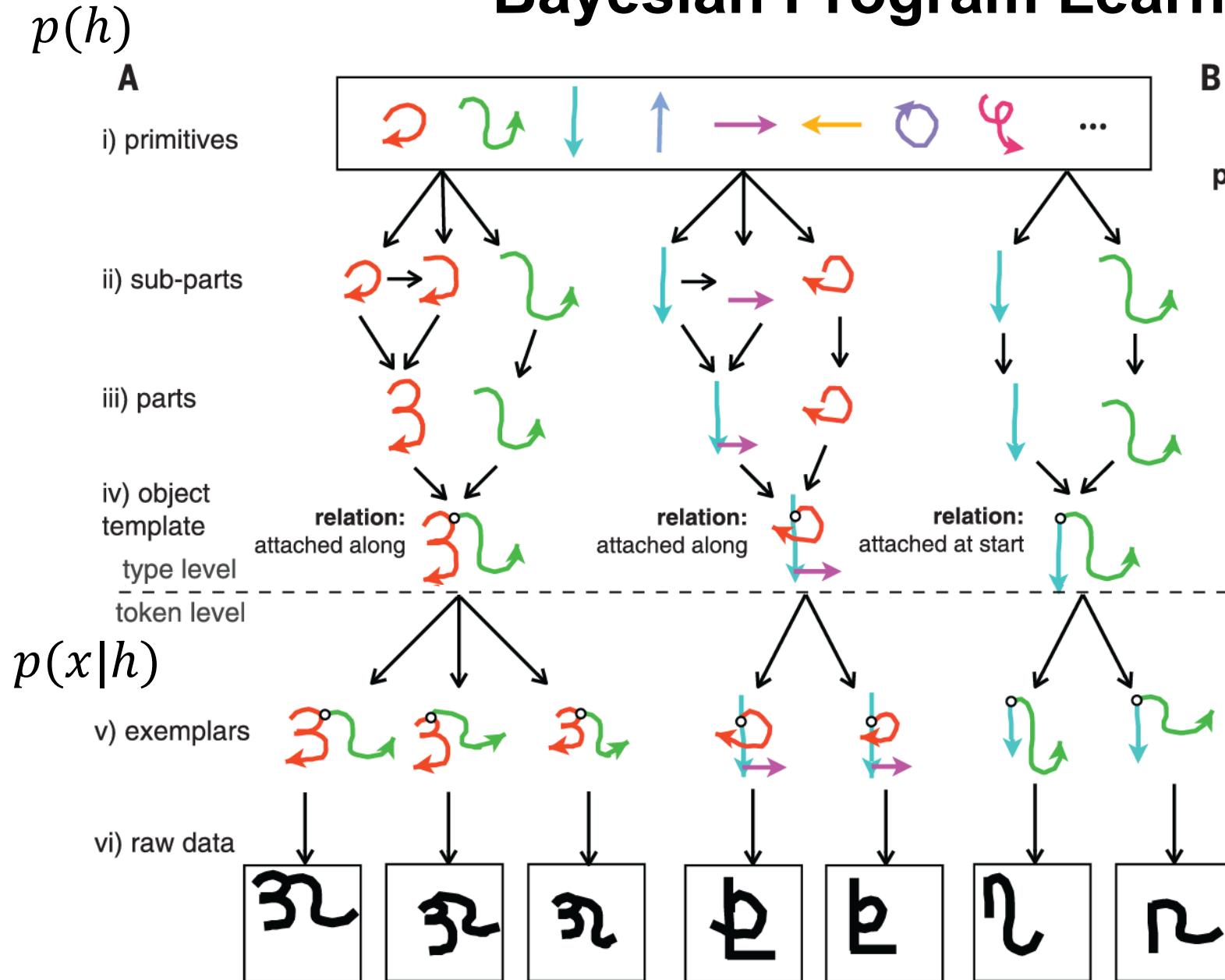


Omniglot challenge

1623 handwritten characters from 50 writing systems

A grid of 16 rows of handwritten characters from various writing systems, including Arabic, Latin, and non-Latin scripts. The characters are arranged in a grid format, showing a variety of scripts and languages.

Bayesian Program Learning (BPL)



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
         $L_i^{(m)} \leftarrow P(L_i^{(m)} | R_i, T_1^{(m)}, \dots, T_{i-1}^{(m)})$   ▷ Sample part's start location
         $T_i^{(m)} \leftarrow f(L_i^{(m)}, S_i^{(m)})$         ▷ 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  $I^{(m)}$ 

```

Image rendering program (with variance)

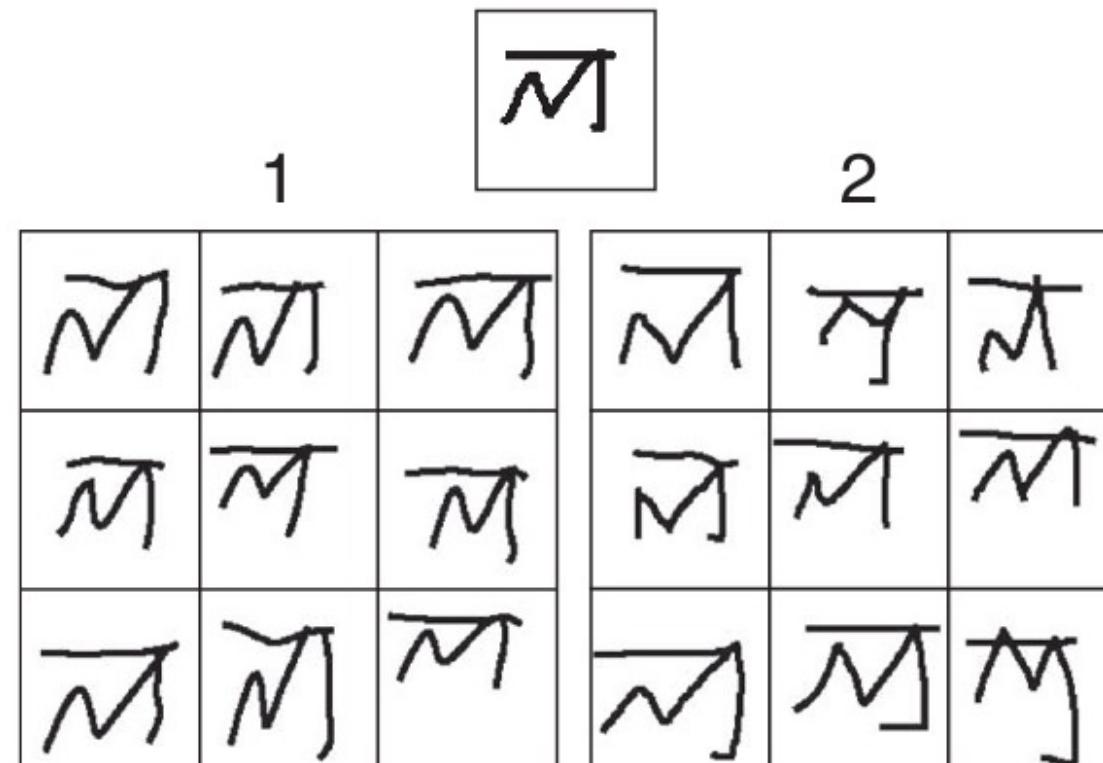
Bayesian Program Learning (BPL)

$$p(x|h)$$

For each pixel, the rendering program gives a Bernoulli distribution



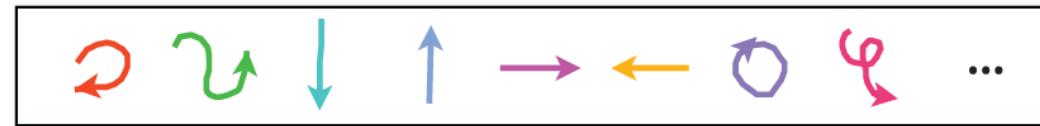
Generating new handwritten examples: A Turing test



Human or Machine?

How to learn to generate programs?

- Primitives?



- Hyper parameters of distributions (e.g., $P(\#parts)$)?

Recap: A toolkit for solving them

- 1. How does abstract knowledge guide learning and inference sparse data?

Bayesian inference in
probabilistic generative models.

$$P(h | d) = \frac{P(d | h)P(h)}{\sum_{h_i \in H} P(d | h_i)P(h_i)}$$

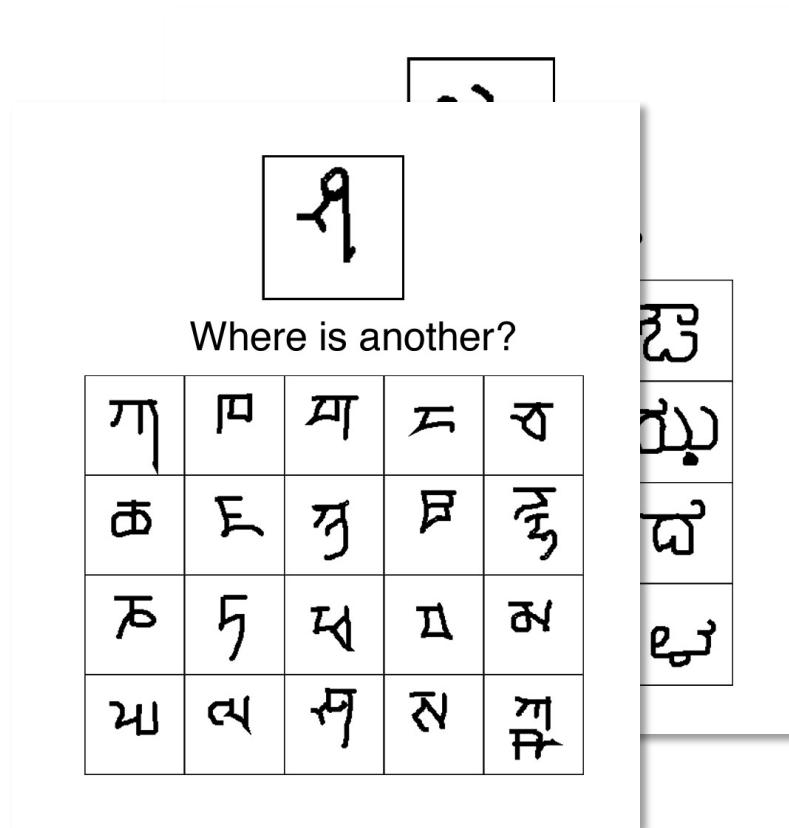
- 2. What form does that knowledge take, across different domains and tasks? Probabilities defined richly structured symbolic representations:
spaces, graphs, grammars, logical predicates, schemas...
- 3. How is that knowledge itself constructed, from some combination of **innate specifications and experience**?

Hierarchical models, with inference at multiple levels.

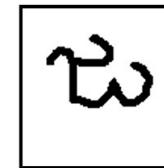
In machine learning terms: learning models as probabilistic inference, “**learning to learn**”/meta-learning, transfer learning, learning representations and learning inductive biases

Learning to learn / meta learning)

Training one-shot classification problems
(with ground-truth labels)



Testing (new) one-shot classification problems



Where is another?

ଅ	ବୁ	ନ୍ତୁ	ଦୁ	ପୁ
କୁ	ନୁ	ଦୁ	ବୁ	ପୁ
ପୁ	ନୁ	ଦୁ	ବୁ	ନ୍ତୁ
ବୁ	ନୁ	ଦୁ	ପୁ	ନ୍ତୁ

5 writing systems (5 one-shot classification problems)

How to learn to generate programs?

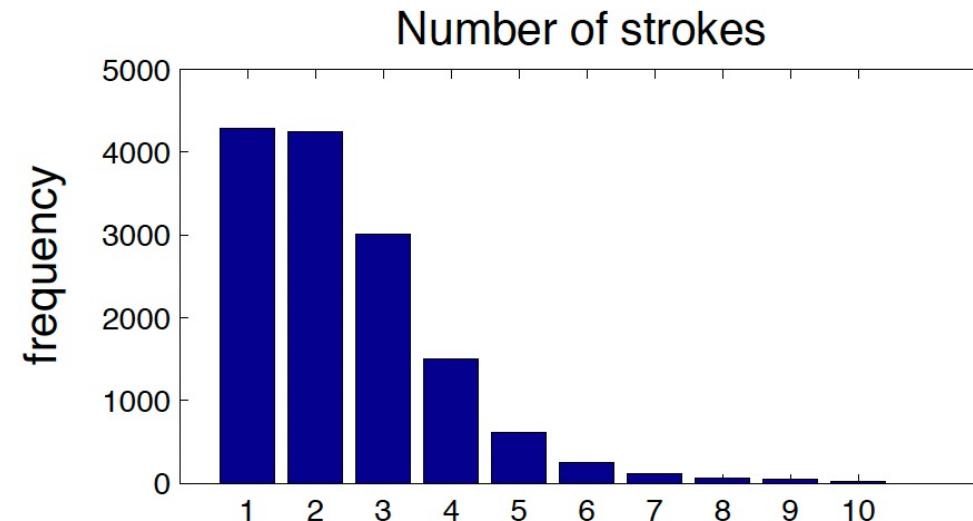
- Primitives?

Clustering segments from training examples

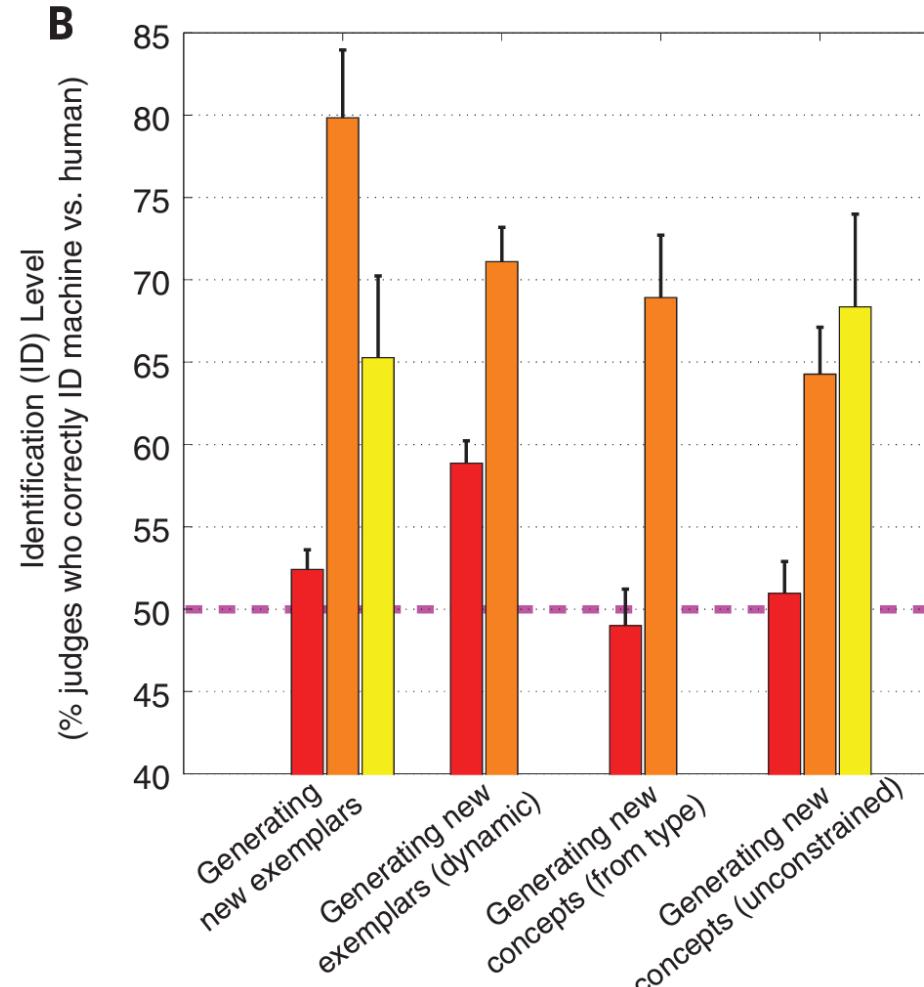
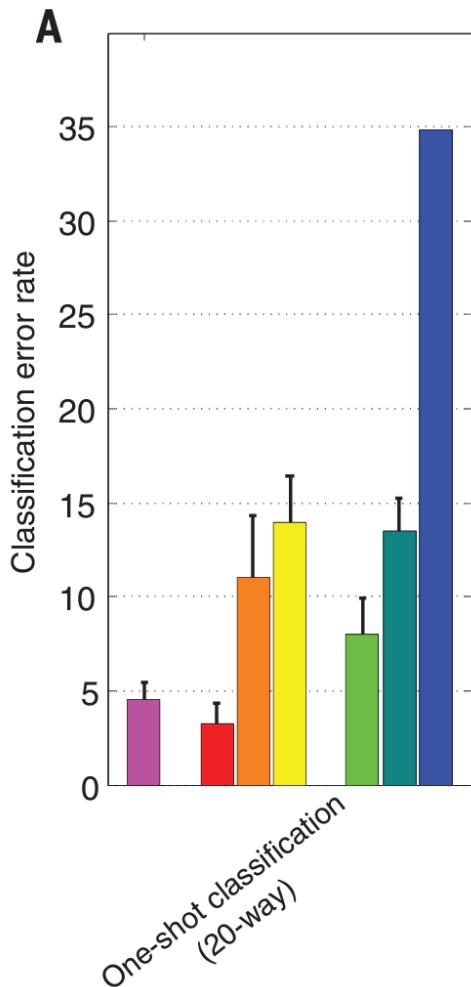
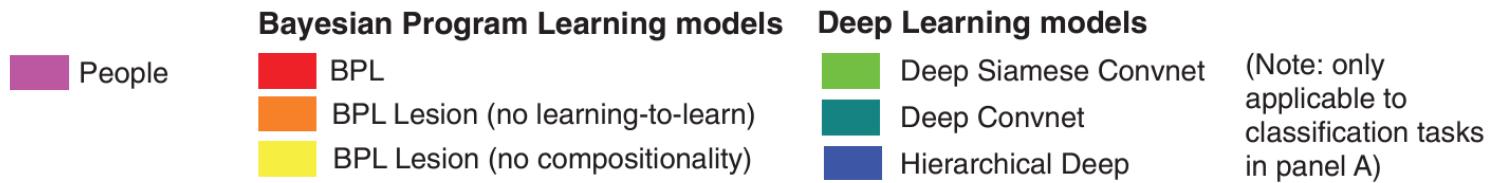


- Hyper parameters of distributions (e.g., $P(\#\text{parts})$)?

Distribution of #parts (#strokes)



Human vs Models



Omniglot challenge

The Omniglot challenge: a 3-year progress report

Brenden M. Lake¹, Ruslan Salakhutdinov

¹Department of Psychology and Center for Dat

²Machine Learning Department, Carn

³Department of Brain and Cognitive Sciences and Cent

	Original	Augmented		
	Within alphabet	Within alphabet (minimal)	Within alphabet	Between alphabet
background set				
# alphabets	30	5	30	40
# classes	964	146	3,856	4,800
2015 results				
Humans	$\leq 4.5\%$			
BPL	3.3%	4.2%		
Simple ConvNet	13.5%	23.2%		
Siamese Net			8.0%*	
2016-2018 results				
Prototypical Net	13.7%	30.1%	6.0%	4.0%
Matching Net				6.2%
MAML				4.2%
Graph Net				2.6%
ARC			1.5%*	2.5%*
RCN	7.3%			
VHE	18.7%			4.8%

What can be learned? Can we use less human inductive biases?

- BPL: learning hyper parameters
- Can we learn more?

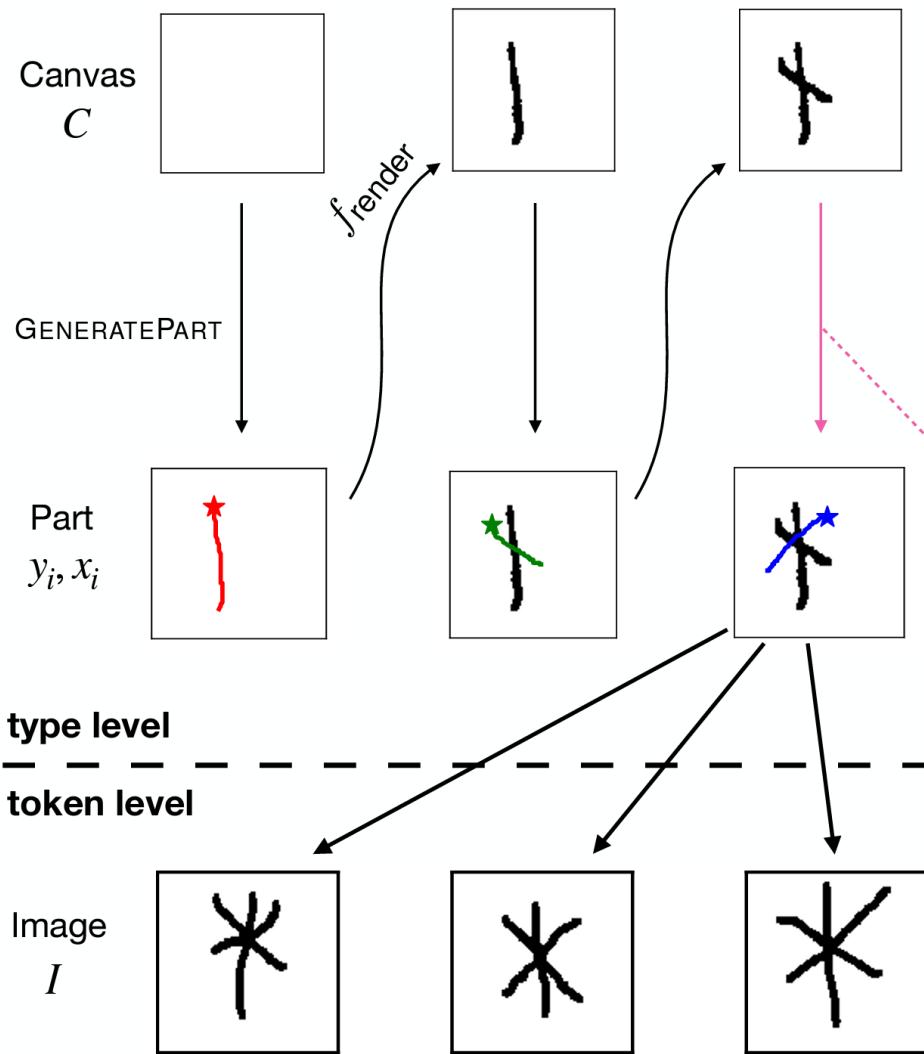
Generative neuro-symbolic modeling (GNS)

Published as a conference paper at ICLR 2021

LEARNING TASK-GENERAL REPRESENTATIONS WITH GENERATIVE NEURO-SYMBOLIC MODELING

Reuben Feinman & Brenden M. Lake
New York University
`{reuben.feinman,brenden}@nyu.edu`

Generative neuro-symbolic modeling (GNS)



procedure GENERATETYPE

$C \leftarrow 0$

while true **do**

$[y_i, x_i] \leftarrow \text{GENERATEPART}(C)$

▷ Initialize blank image canvas

$C \leftarrow f_{\text{render}}(y_i, x_i, C)$

▷ Sample part location & parameters

$v_i \sim p(v | C)$

▷ Render part to image canvas

if v_i **then**

break

$\psi \leftarrow \{\kappa, y_{1:\kappa}, x_{1:\kappa}\}$

return ψ

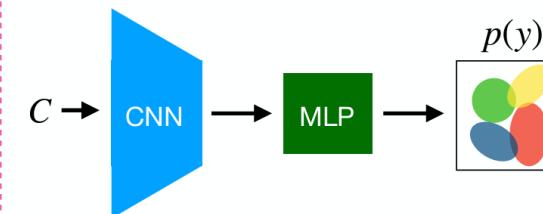
▷ Sample termination indicator

▷ Terminate sample

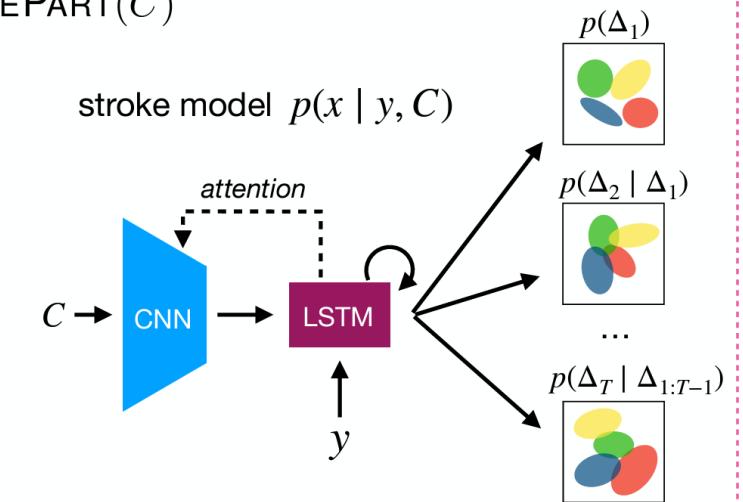
▷ Return concept type

GENERATEPART(C)

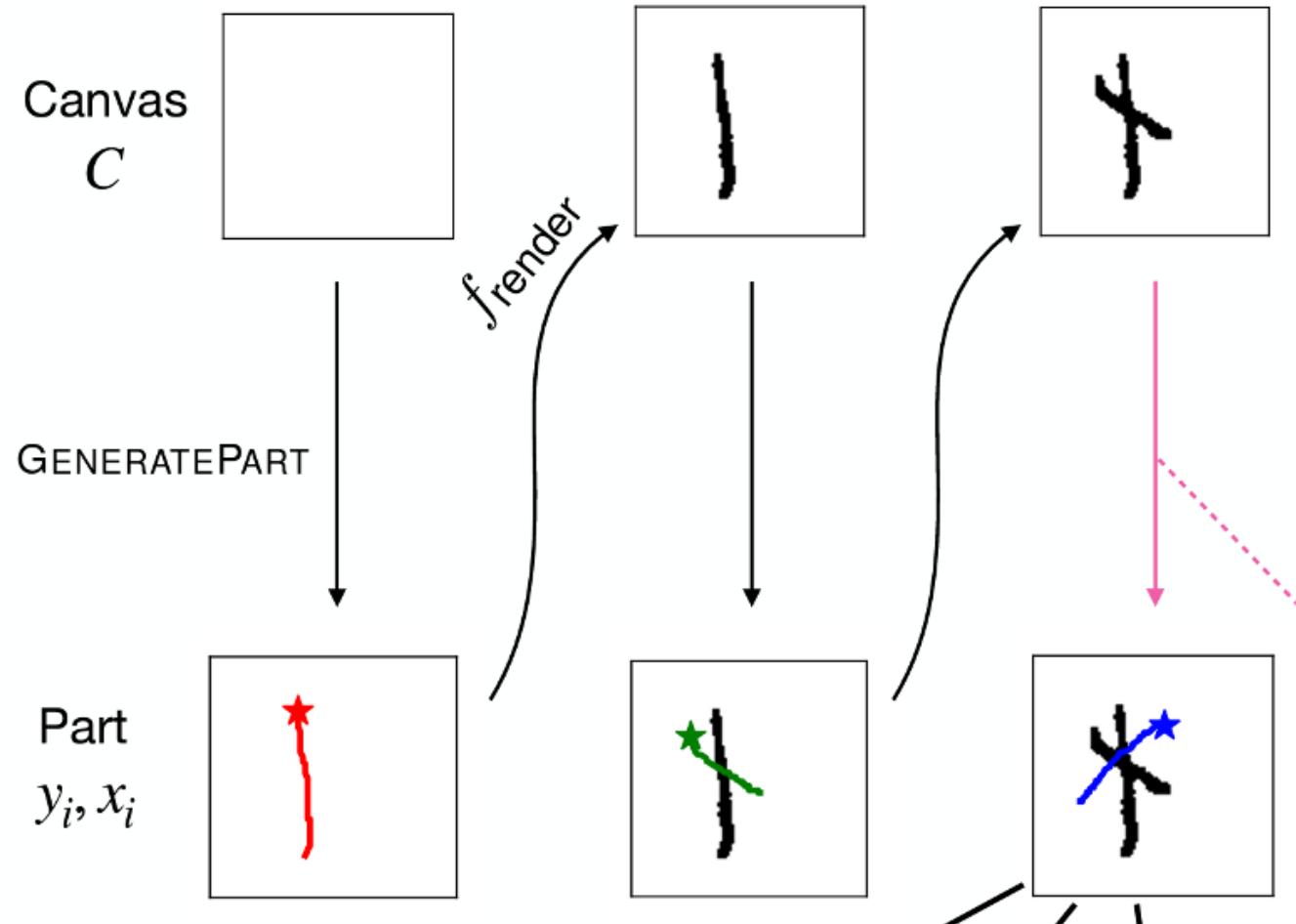
location model $p(y | C)$



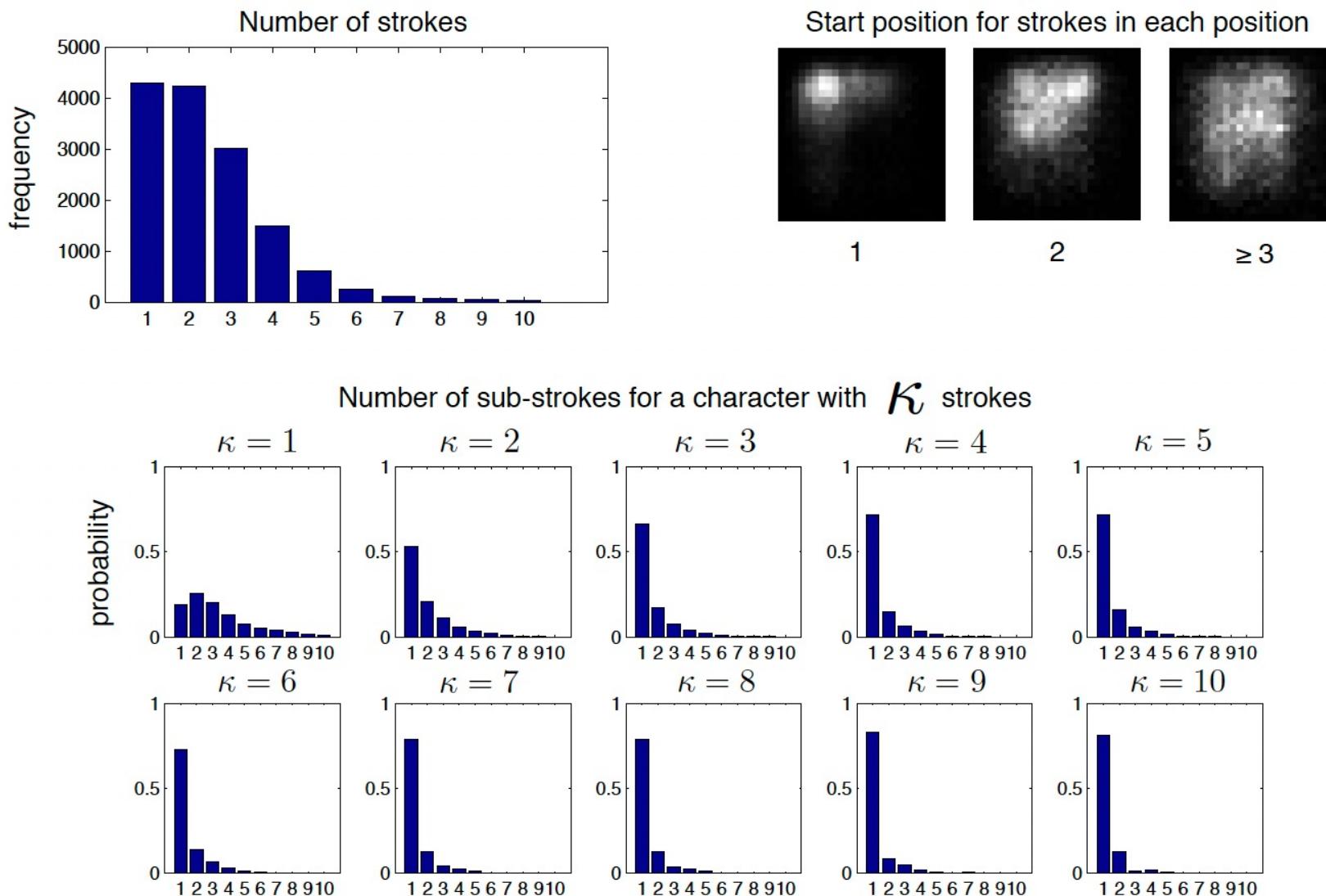
stroke model $p(x | y, C)$



Neural networks for learning to generate types



In comparison to BPL



Generative neuro-symbolic modeling (GNS)

- Inductive bias (**symbolic**): the structure of type/concept generation programs (there are parts and they need to be connected), symbolic rendering programs
- Learning what kinds of parts and relationships can be used to synthesize programs (without human priors) using **neural networks**

Table 2: Test error on within-alphabet one-shot classification.

Model	Error
GNS	5.7%
BPL [29]	3.3%
RCN [12]	7.3%
VHE [21]	18.7%
Proto. Net [45]	13.7%
ARC [44]	1.5%*

*used 4x training classes

More domains?

- Given a domain with multiple concepts, come up with a DSL and write programs to interpret different concepts (concept = program)

DreamCoder: Growing generalizable, interpretable knowledge with wake-sleep Bayesian program learning

Kevin Ellis,^{1,4,5} Catherine Wong,^{1,4,5} Maxwell Nye,^{1,4,5}
Mathias Sablé-Meyer,^{1,3} Luc Cary,¹ Lucas Morales,^{1,4,6} Luke Hewitt,^{1,4,5}
Armando Solar-Lezama,^{1,2,6} Joshua B. Tenenbaum^{1,2,4,5}

¹MIT ²CSAIL ³NeuroSpin ⁴Center for Brains, Minds, and Machines

⁵Department of Brain and Cognitive Sciences ⁶Department of Electrical Engineering and Computer Science

List Processing

Sum List
 $[1 \ 2 \ 3] \rightarrow 6$
 $[4 \ 6 \ 8 \ 1] \rightarrow 17$

Double
 $[1 \ 2 \ 3] \rightarrow [2 \ 4 \ 6]$
 $[4 \ 5 \ 1] \rightarrow [8 \ 10 \ 2]$

Check Evens
 $[0 \ 2 \ 3] \rightarrow [T \ T \ F]$
 $[2 \ 9 \ 6] \rightarrow [T \ F \ T]$

Text Editing

Abbreviate
Allen Newell → A.N.
Herb Simon → H.S.

Drop Last Three
shrdlu → shr
shakey → sha

Extract
a b (c) → c
a (bee) see → see

Regexes

Phone numbers
(555) 867-5309
(650) 555-2368

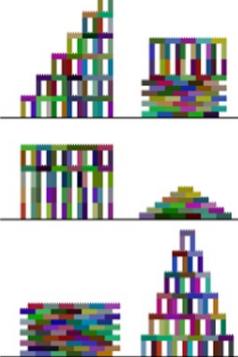
Currency
\$100.25
\$4.50

Dates
Y1775/0704
Y2000/0101

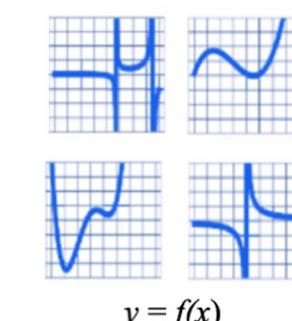
LOGO Graphics



Block Towers



Symbolic Regression



Recursive Programming

Filter Red
[■■■■■■] → [■■■]
[■■■■■■■■] → [■■■■■■]
[■■■■■■■■■] → [■■■■■■]

Length
[■■■■■■] → 4
[■■■■■■■■] → 6
[■■■■] → 3

Physical Laws

$$\vec{a} = \frac{1}{m} \sum_i \vec{F}_i$$

$$\vec{F} \propto \frac{q_1 q_2}{|\vec{r}|^2} \hat{r}$$

$$R_{\text{total}} = \left(\sum_i \frac{1}{R_i} \right)^{-1}$$

Abstraction and Reasoning Corpus (ARC) Dataset

Inspired by humans' **core knowledge system** (Spelke & Kinzler, 2007)

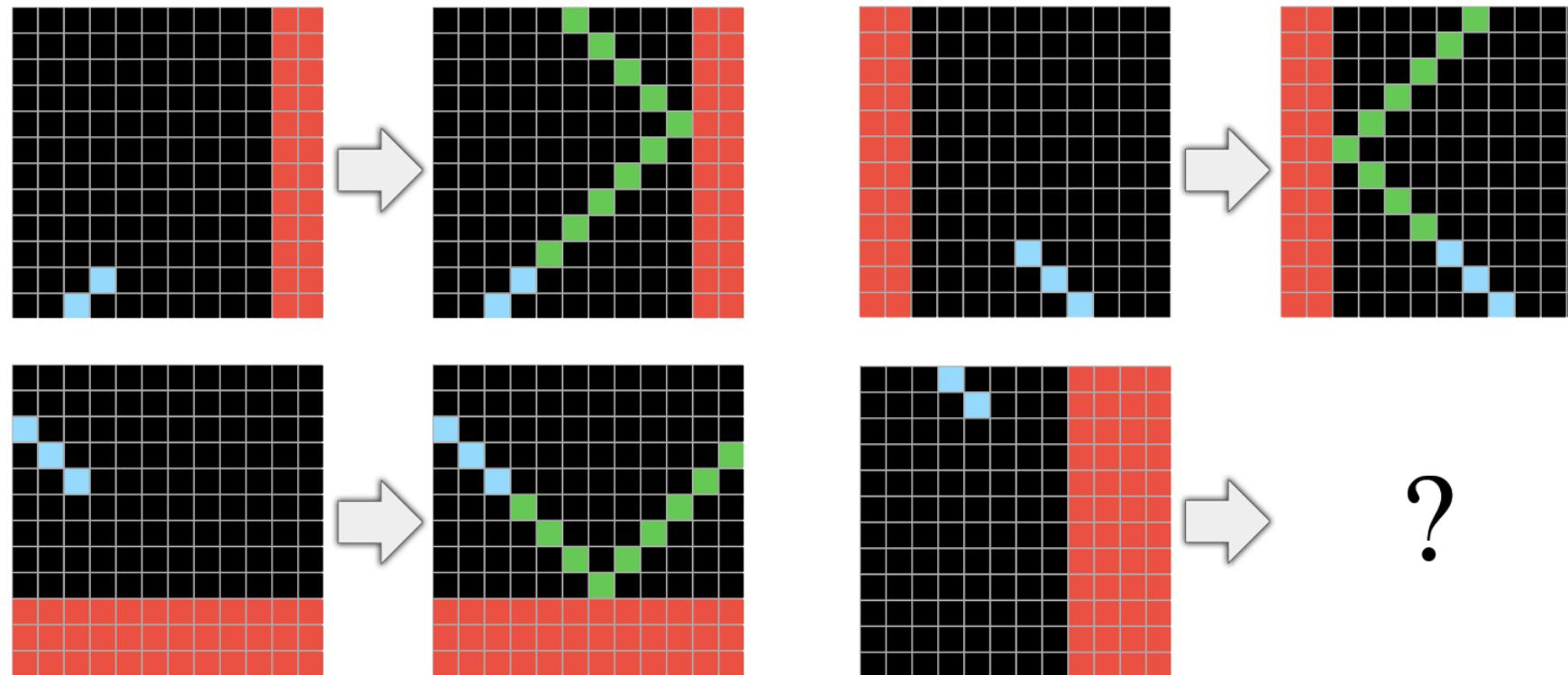


Figure 8: A task where the implicit goal is to extrapolate a diagonal line that “rebounds” upon contact with a red obstacle.

Few-shot instruction learning

Human few-shot learning of compositional instructions

Brenden M. Lake^{1,2}, Tal Linzen³, and Marco Baroni^{2,4}

¹New York University, ²Facebook AI Research, ³John Hopkins University, ⁴ICREA

Training		Testing	
Primitives	Function 3	Function 1	Function compositions
dax ●	wif ●	lug kiki wif ● ●	zup fep ● ● ● 88% zup fep kiki lug ● ● ● ● 85%
lug ●	zup ●	dax kiki lug ● ●	Function 2
			zup blicket lug ● ● ● 79% wif kiki zup fep ● ● ● ● 85%
Function 1	Function compositions	Function 3	lug kiki wif blicket zup ● ● ● ● 65% zup blicket wif kiki dax fep ● ● ● ● ● ● 70%
lug fep ● ● ●	lug fep kiki wif ● ● ● ●	zup blicket zup ● ● ● 88% zup blicket zup kiki zup fep ● ● ● ● ● ● 75%	
dax fep ● ● ●	wif kiki dax blicket lug ● ● ● ●	zup kiki dax ● ● 86%	
Function 2	lug kiki wif fep ● ● ● ●	wif kiki zup ● ● 86%	
lug blicket wif ● ● ●	wif blicket dax kiki lug ● ● ● ●		
wif blicket dax ● ● ●			

Can we be even more flexible?

- Natural languages can flexibly describe any concepts
- Reasoning with language?
- Let's revisit the number game!

Reasoning with language

Let's play a game. I have programmed a computer to pick out numbers according to whether they fit some simple criterion. All numbers between 1-100 either "yes" or "no" numbers; "yes" means the number fits. I won't tell you the program I wrote, but I will tell you several numbers between 1 and 100 that are all examples of "yes" numbers.

After I give you these examples, you have to respond with a list of any other numbers in the 1-100 range that you think are also "yes" numbers. Please also include at least ten other numbers that you think are "no" numbers or might be on the borderline of "yes" or "no". Order your list in order of your confidence that a given number is a "yes" number, and after each number, give a rating of 1-7 expressing your confidence: 7 means you are absolutely sure that the number is a "yes" number; 1 means you are absolutely sure that the number is a "no" number; 4 means it's a 50-50 bet, could go either way whether the number is "yes" or "no"; and intermediate numbers express intermediate degrees of confidence. Ready to play?



GPT-4

Absolutely! I'm ready to play. Please provide the list of "yes" numbers as examples, and I'll do my best to determine the pattern and provide my guesses.

Here are four examples: 60, 80, 10 and 30



GPT-4

Alright, based on the numbers you've given me (10, 30, 60, and 80), it seems that all of these numbers end in a 0. Given this, I suspect that the criterion is that the number must be a multiple of 10.

Based on this hypothesis, here's my list:

Yes numbers (with confidence level):

1. 20 (7) - Absolutely sure it's a "yes" number.
2. 40 (7) - Absolutely sure it's a "yes" number.
3. 50 (7) - Absolutely sure it's a "yes" number.
4. 70 (7) - Absolutely sure it's a "yes" number.
5. 90 (7) - Absolutely sure it's a "yes" number.
6. 100 (7) - Absolutely sure it's a "yes" number.

No/borderline numbers (with confidence level):

1. 9 (1) - Absolutely sure it's a "no" number.
- 2. 15 (2) - Fairly sure it's a "no" number.
- 3. 33 (1) - Absolutely sure it's a "no" number.
- 4. 45 (2) - Fairly sure it's a "no" number.
5. 58 (1) - Absolutely sure it's a "no" number.
6. 62 (1) - Absolutely sure it's a "no" number.
7. 77 (1) - Absolutely sure it's a "no" number.
8. 86 (1) - Absolutely sure it's a "no" number.
9. 94 (1) - Absolutely sure it's a "no" number.
- 0. 55 (4) - 50-50 bet. This is my borderline guess since it's the middle of two "yes" numbers (50 and 60).

How did I do?

Can you give a few other borderline or intermediate cases, along with your degree of confidence in them on a scale of 1-7?



GPT-4

Of course! Based on the current hypothesis, I'll try to identify a few numbers that might be considered borderline or intermediate cases:

1. **25 (3)** - It's halfway between two "yes" numbers (20 and 30), so I'm leaning towards it being a "no" number but with some uncertainty.
2. **35 (3)** - Similar reasoning to 25, it's halfway between 30 and 40.
3. **65 (3)** - Positioned between 60 and 70, making it a potential borderline case.
4. **5 (4)** - It's a multiple of 5, but not 10. It's the lowest number divisible by 5 within the range, so it's a 50-50 bet.
5. **95 (3)** - It's close to 100 and is between 90 and 100, making it a potential borderline number.
6. **85 (3)** - It's between 80 and 90, another number in the mid-way point.
7. **75 (3)** - Same reasoning as 65, 35, and 25; it's halfway between 70 and 80.

These are all numbers that are in-between the clear "yes" numbers. The confidence level leans towards them being "no" numbers but not with absolute certainty due to their positioning.

Can you explain the basis for your answers, and your degrees of confidence?



GPT-4

Of course! Let's break down the rationale:

Basis for Answers:

1. The initial numbers you provided were 10, 30, 60, and 80. These numbers all end in '0', suggesting that the pattern might be numbers divisible by 10. This was the primary pattern I discerned from the initial examples.
2. Given this suspected pattern, numbers divisible by 10 within the 1-100 range were classified as "yes" numbers.

Degrees of Confidence Explanation:

1. **For Clear "Yes" Numbers (7):** Numbers like 20, 40, 50, 70, 90, and 100 fit the divisible by 10 pattern perfectly. Hence, I'm entirely confident these are "yes" numbers.

4. For True 50-50 Bet Numbers (4): Number 55 was the epitome of this category. It's right in the middle of 50 and 60. While the initial pattern suggests it's a "no" number, its position made me consider it might be a wildcard. Hence, a confidence level of 4, indicating genuine uncertainty.

Numbers, their positioning between clear "yes" numbers adds a bit of uncertainty. Hence, I'm leaning towards them being "no" numbers but with some doubt, so a confidence level of 3.

4. **For True 50-50 Bet Numbers (4):** Number 55 was the epitome of this category. It's right in the middle of 50 and 60. While the initial pattern suggests it's a "no" number, its position made me consider it might be a wildcard. Hence, a confidence level of 4, indicating genuine uncertainty.

The key challenge here is working with limited examples. The initial pattern of multiples of 10 seemed the most evident. However, with more examples or iterations of the game, I could refine the pattern and improve the accuracy of my guesses and confidence levels.

Let's play a game. I have programmed a computer to pick out numbers according to whether they fit some simple criterion. All numbers between 1-100 either "yes" or "no" numbers; "yes" means the number fits. I won't tell you the program I wrote, but I will tell you several numbers between 1 and 100 that are all examples of "yes" numbers.

After I give you these examples, you have to respond with a list of any other numbers in the 1-100 range that you think are also "yes" numbers. Please also include at least ten other numbers that you think are "no" numbers or might be on the borderline of "yes" or "no". Order your list in order of your confidence that a given number is a "yes" number, and after each number, give a rating of 1-7 expressing your confidence: 7 means you are absolutely sure that the number is a "yes" number; 1 means you are absolutely sure that the number is a "no" number; 4 means it's a 50-50 bet, could go either way whether the number is "yes" or "no"; and intermediate numbers express intermediate degrees of confidence. Ready to play?

Here are two examples of "yes" numbers for the program: 60 and 80.



GPT-4

Alright! Given the examples you provided, it seems that we're working with the numbers 60 and 80 as "yes" numbers. The most immediate pattern I can observe is that both numbers are divisible by 20. I'll hypothesize that all numbers divisible by 20 in the range 1-100 might be "yes" numbers.

Based on this hypothesis, here's my list:

1. **20** - Confidence Level: 7 (Based on my initial hypothesis, I'm quite confident about this.)
2. **40** - Confidence Level: 7 (Continuing the trend.)
3. **100** - Confidence Level: 7 (It's the next in line after 80.)

I'm also considering the mid-way numbers between the given "yes" numbers to see if there's any trend with them:

I'm also considering the mid-way numbers between the given "yes" numbers to see if there's any trend with them:

1. **70** - Confidence Level: 5 (It's between 60 and 80, but I'm less certain given it's not divisible by 20.)
2. **50** - Confidence Level: 5 (Again, not divisible by 20 but is halfway between 40 and 60.)

Now, for the "no" numbers or borderline ones:

1. **61** - Confidence Level: 4 (Just a number close to 60, not sure if it has any significance.)
2. **79** - Confidence Level: 4 (Close to 80 but not sure of its relevance.)
3. **30** - Confidence Level: 3 (It is divisible by 20 but it's below 60 so I'm less certain.)
4. **10** - Confidence Level: 2 (It's the first number divisible by 20, but being the first makes me think it might not fit.)
5. **85** - Confidence Level: 2 (It's close to 80, but doesn't seem to fit the pattern.)
6. **5** - Confidence Level: 1 (I'm quite certain it's a "no" based on the examples given.)
7. **33** - Confidence Level: 1 (Randomly chosen, seems not to fit the potential pattern.)
8. **91** - Confidence Level: 1 (Again, randomly chosen and doesn't fit the pattern.)
9. **17** - Confidence Level: 1 (Seems like a "no" number based on the pattern.)
10. **98** - Confidence Level: 1 (Close to 100 but doesn't seem relevant based on the pattern.)

This is a fun game! Let me know how I did!

Let's play a game. I have programmed a computer to pick out numbers according to whether they fit some simple criterion. All numbers between 1-100 either "yes" or "no" numbers; "yes" means the number fits. I won't tell you the program I wrote, but I will tell you several numbers between 1 and 100 that are all examples of "yes" numbers.

After I give you these examples, you have to respond with a list of any other numbers in the 1-100 range that you think are also "yes" numbers. Please also include at least ten other numbers that you think are "no" numbers or might be on the borderline of "yes" or "no". Order your list in order of your confidence that a given number is a "yes" number, and after each number, give a rating of 1-7 expressing your confidence: 7 means you are absolutely sure that the number is a "yes" number; 1 means you are absolutely sure that the number is a "no" number; 4 means it's a 50-50 bet, could go either way whether the number is "yes" or "no"; and intermediate numbers express intermediate degrees of confidence. Ready to play?

Here are two examples of "yes" numbers for the program: 13 and 47.



Alright, given the examples 13 and 47, let's try to discern a pattern. Both numbers are prime, so one hypothesis could be that "yes" numbers are prime numbers.

Let's run with this hypothesis for now.

Numbers I think are "yes" numbers:

1. 2 - Confidence: 7 (It's the smallest prime number)
2. 3 - Confidence: 7
3. 5 - Confidence: 7
4. 7 - Confidence: 7

5. 11 - Confidence: 7
6. 17 - Confidence: 7
7. 19 - Confidence: 7
8. 23 - Confidence: 7
9. 29 - Confidence: 7
10. 31 - Confidence: 7

(Continuing the list would include other prime numbers up to 100, but I'll stop here for brevity.)

Numbers that might be "no" numbers or on the borderline:

1. 4 - Confidence: 2 (It's an even number but not prime.)
2. 6 - Confidence: 2
3. 8 - Confidence: 2
4. 9 - Confidence: 2 (It's the square of a prime, which might be interesting.)
5. 10 - Confidence: 2
6. 12 - Confidence: 2
7. 14 - Confidence: 2 (Even though it's close to 13, it's not prime.)
8. 15 - Confidence: 3 (It's odd but not prime. Given our limited data, it's harder to be sure about this one.)
9. 16 - Confidence: 2
10. 18 - Confidence: 2

It's worth noting that this is only one possible hypothesis based on the two numbers provided. The actual criterion could be something different entirely.