Grounding Compositional Hypothesis Generation in Specific Instances

Neil R. Bramley[1] (neil.bramley@nyu.edu), **Anselm Rothe**[1] (anselm@nyu.edu)
Joshua B. Tenenbaum[2] (jbt@mit.edu), **Fei Xu**[3] (fei_xu@berkeley.edu), **Todd M. Gureckis**[1] (gureckis@nyu.edu)
[1]Department of Psychology, NYU, New York, NY, [2]Brain & Cognitive Sciences, MIT, Cambridge, MA, [3]Department of Psychology, UC Berkeley, CA

Abstract

A number of recent computational models treat concept learning as a form of probabilistic rule induction in a space of language-like, compositional concepts. Inference in such models frequently requires repeatedly sampling from a (infinite) distribution over possible concept rules and comparing their relative likelihood in light of current data or evidence. However, we argue that most existing algorithms for top-down sampling are inefficient and cognitively implausible accounts of human hypothesis generation. As a result, we propose an alternative, Instance Driven Generator (IDG), that constructs bottom-up hypotheses directly out of encountered positive instances of a concept. Using a novel rule induction task based on the children's game Zendo, we compare these "bottom-up" and "top-down" approaches to inference. We find that the bottom-up IDG model accounts better for human inferences and results in a computationally more tractable inference mechanism for concept learning models based on a probabilistic language of thought.
Keywords: discovery; program induction; probabilistic language of thought, active learning; hypothesis generation

A number of recent papers argue that people come to learn abstract hypotheses about the world through inference in a "probabilistic language of thought" (Goodman, Tenenbaum, & Gerstenberg, 2015). That is, a system of conceptual primitives and stochastic rules for how they can be combined. Such models capture the fact that human thinking exhibits language-like compositionality and systematicity (Fodor, 1987; Lake, Salakhutdinov, & Tenenbaum, 2015), allowing us to combine and repurpose simple concepts to construct richer ones (Goodman, Tenenbaum, Feldman, & Griffiths, 2008; Piantadosi, Tenenbaum, & Goodman, 2016). In order to model inference in such models it is usually assumed that people approximate a posterior distribution over possible abstract expressions using stochastic search algorithms (Hastings, 1970). However, such approaches usually require the top-down generation of very large numbers of samples to stand a good chance of including the true hypothesis making them implausible algorithmic level accounts of human learning and hypothesis generation (Lewis, Perez, & Tenenbaum, 2014; Schulz, 2012).

However, assuming humans are endowed with some type of language-like representation for forming abstract hypotheses, one would assume that this language can be of use not only for inferring abstract hypotheses or ideas but also to describe concrete states of affairs in the world and specific instances. For example, looking at a visual scene one can describe various facts that hold (e.g., "the coffee cup is on top of the table", "the window is closed"). Such observations are essentially mental statements about the "data" one is perceiving. Given this ability, it seems it would be useful for a symbolic inferential hypothesis system to use these descriptions to help "seed" the hypothesis generation and search process. So for instance instead of starting from scratch and sampling repeatedly from a prior distribution, a learner might first start by considering hypotheses that match some of all of the factual statements considered about the data/world. By exploring potential generalizations and restrictions on those abstract hypotheses, one can more efficiently arrive at some deeper abstract causal knowledge (e.g., did the person leave their office in a hurry?) because each hypothesis is, at minimum, consistent with known data.

In this paper, we explore a middle ground between top-down sample-driven and bottom-up instance-driven concept learning. We propose a instance driven grammatical hypothesis generator, and demonstrate that it provides a closer characterization of human generalizations than a fully top-down approach, as well as being more computationally efficient.

Discovery learning in Zendo

To explore this idea, we developed a new task environment that, while formally specified, is open ended enough to provide a challenging and naturalistic test bed for concept learning. The task is inspired by the tabletop scientific learning game Zendo^{TM}. In it, learners both observe and create their own *scenes*, which are arrangements of 2D triangular objects called *cones* on a flat surface (as depicted in Figure 1). The goal is to identify a hidden rule that divides scenes into *rule-following* and *not rule-following* scenes. Scenes could contain a varied number of cones, each of which had two immutable properties: size$\in$ {small, medium, large} and color$\in$ {red, green, blue} as well as having continuous scene-specific x$\in$(0,8), y$\in$(0,6) positions and orientations$\in(0,2\pi)$[1]. The scenes are subject to gravity meaning there are familiar physical constraints on how they might be arranged. In addition to cones' immutable and positional properties, scenes also admit many complex properties arising from the relative features and arrangement of different cones. For instance, subsets of cones might share a feature value (i.e., be the same color, or have the same orientation) or be ordered on another (i.e., be larger than, or above) and pairs of cones might have properties like pointing at one another or touching. This results in a rich space of potential concepts that require an expressive conceptual language. Following Piantadosi et al. (2016), we assume the true latent space of possible concepts in our task are those expressible in first order logic combined with lambda abstraction and full knowledge of the potentially relevant features of the scene. Lambda abstraction provides a simple general formalism for binding entities to variables (Church, 1932). For the current context, binding sets of cones

[1]We round these to one decimal place in evaluating rules to allow for perceptual uncertainty.

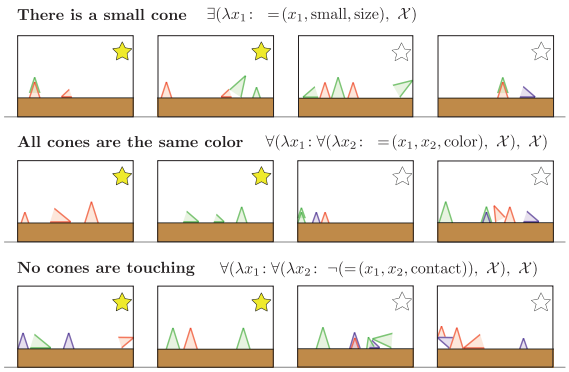

Figure 1: Three examples of rules in our task in words and lambda calculus with example scenes below. Yellow stars indicate scenes that follow the rule, white stars indicate scenes that do not.

Table 1: λ-abstraction Based Probabilistic Grammar.

Productions				
Start	$S \rightarrow$	$\exists(\lambda x_i: A, \mathcal{X})$	$\forall(\lambda x_i: A, \mathcal{X})$	$N_I(\lambda x_i: A, J, \mathcal{X})$
Bind additional	$A \rightarrow$	B	S	
Expand	$B \rightarrow$	C	$H(B,B)$	$\neg(B)$
Function	$C \rightarrow$	$=(x_i, D1)$	$I(x_i, D2)$	$=(x_i, x_j, E1)$[a]
		$I(x_i, x_j, E2)$[a]	$\Gamma(x_i, x_j, E3)$[a]	
Feature/value	$D1 \rightarrow$	value[b] ,	feature	
(numeric only)	$D2 \rightarrow$	value[b] ,	feature	
Feature	$E1 \rightarrow$	feature		
(numeric only)	$E2 \rightarrow$	feature		
(relational)	$E3 \rightarrow$	feature		
Boolean	$H \rightarrow$	$\wedge$	$\vee$	...
Inequality	$I \rightarrow$	$\leq$	$\geq$	$>$
		$<$		
Number	$J \rightarrow$	$n \in Z_5$		

Note: Each production rule maps a capital letter (column 2) to several possible replacements (right hand columns). Context-sensitive aspects of the grammar: [a]Bound variable(s) sampled uniformly without replacement from set; expressions requiring multiple variables censored if only one. [b] The value is always sampled uniformly from the support of the feature selected in D.

to different variables allows our grammar to assert relations between distinct subsets of the cones in a scene.

Top-down, "guess and check" inference mechanisms

As mentioned above, one solution to the problem of learning within an infinite hypothesis space is to sample hypotheses by composing them stochastically from an underlying grammar of sufficient expressivity. Here, we consider a grammar (specifically a *probabilistic context free grammar* or PCFG; Ginsburg, 1966) that can be used to produce any rule that can be expressed with first-order logic and lambda abstraction. When set up correctly, "simpler concepts" (i.e., those composed of fewer parts, cf. Feldman, 2000), will have a naturally higher prior probability of being produced and so will be favored over more complex ones equally able to explain the data. This is necessary since the setting ensures there will always still be an infinite number of potential rules consistent with any data (Gold et al., 1967).

With a PCFG, each hypothesis begins life as a string containing a single *non-terminal symbol* (here, S) that is replaced using rewrite rules, or *productions*. These productions are repeatedly applied to the string, replacing non-terminal symbols with a mixture of other non-terminal symbols and terminal fragments of first order logic, until no non-terminal symbols remain. The productions are so designed that the resulting string is guaranteed to be a valid grammatical expression. In addition, by having the productions tie the expression to bound variables and truth statements, our PCFG serves as an automatic concept generator. Table 1 details the PCFG we consider in the current paper and Figure 2a gives two example PCFG samples. We use capital letters as non-terminal symbols and each rewrite is sampled from the available productions for a given symbol.[2] Because some of the productions involve branching (e.g., $B \rightarrow H(B,B)$, see Table 1), the resultant string can become arbitrarily long and complex, involving multiple boolean functions and complex relationships between bound variables. The probabilities for each production in a PCFG can be fit to human judgments, and different PCFGs containing different primitives and expansions can be compared. In this way, recent work has attempted to infer the logical primitives of thought (Goodman et al., 2008; Piantadosi et al., 2016).

What these PCFG approaches have in common is that they provide a generative mechanism for sampling from a prior over all possible concepts. However, these sampled "guesses" must then be tested against data. Unfortunately, many samples, even from a well tuned PCFG are likely to be tautological (i.e., "All cones are red or not red"), contradictory (i.e., "There is a cone that is red and not red"), physically impossible ("There are two distinct objects have the same position and orientation") or simply inconsistent with whatever data a learner has already encountered. Indeed, around 20% of the hypotheses generated by the grammar in Table 1 are tautologies, and 15% are contradictions. For these reasons, the procedure is inherently inefficient, and typically requires a very large numbers of samples in order to reliably provide non-trivial rules.

Instance driven hypothesis generation

Our Instance Driven Generation (IDG) proposal is related to the PCFG idea but with one major difference. Rather than generating guesses entirely stochastically, before checking them against the data, we propose that people generate guesses *inspired* by an encountered positive example (cf. Michalski, 1969). Concretely, we propose that learners start by observing the features of objects in a rule-following scene and use these to back out a true logical statement about the scene in a stochastic but truth preserving way. In this way the learner does not generate uniformly from all possible logical statements, but directly from the restricted space of statements true of the current observation. Figure 2b motivates this approach. Here a learner begins their hypothesis generation with an observation of a scene that follows the hidden rule. To generate hypotheses as candidates for the hidden rule, we assume the learner uses the following procedure:

[2]The grammar is not strictly context free because the bound variables (x_1, x_2, etc.) are shared across contexts (e.g. x_1 is evoked twice in both expressions generated in Figure 2a).

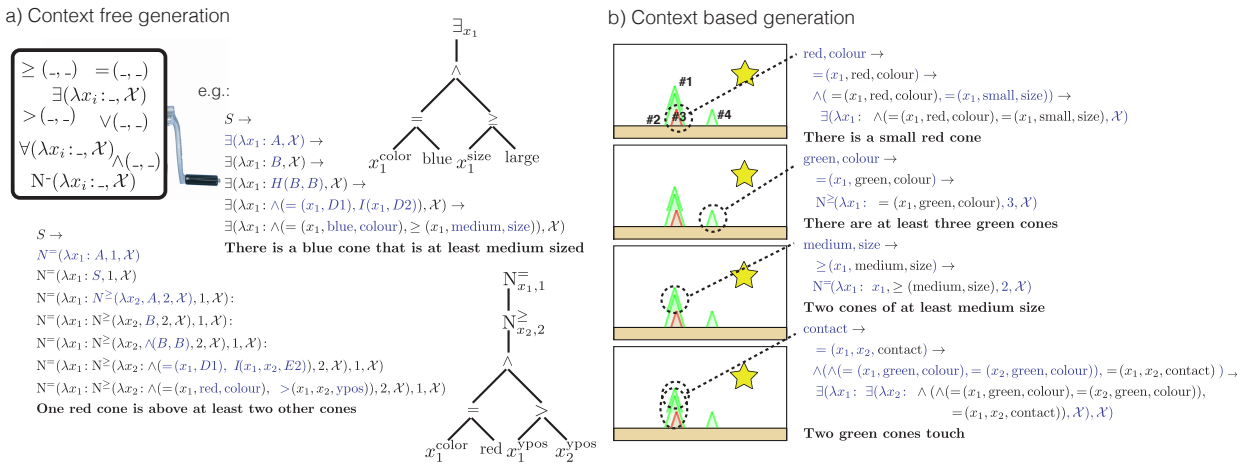

Figure 2: a) Example generation of hypotheses using PCFG in Table 1. b) Examples of IDG hypothesis generation based on an observation of a scene that follows the rule. New additions on each line are marked in blue.

1. **Observe.** With uniform probability, either:

   (a) Sample a cone from the observation, then sample one of its features — e.g., #1:[3] "medium, size" or {#3}: "red, color".

   (b) Sample two cones uniformly without replacement from the observation, and samples any shared or pairwise feature — e.g., {#1,#2}: "size", or "contact"

2. **Functionize.** Bind a variable for each sampled cone in Step 1 and sample a true (in)equality statement relating the variable(s) and feature:

   (a) For a statement involving an unordered feature there is only one possibility — e.g, {#3}: "$=(x_1,\text{red},\text{color})$", or for {#1,#2}: "$=(x_1,x_2,\text{color})$"

   (b) For a single cone and an ordered feature this could also be a nonstrict inequality ($\geq$ or $\leq$). We assume a learner only samples an inequality if it expands the number of cones picked out from the scene relative to an equality — e.g., in Figure 2b, there is also a large cone {#1} so either $\geq(x_1,\text{medium},\text{size})$ or $=(x_1,\text{medium},\text{size})$ might be selected.

   (c) For two cones and an ordered feature, either strict or non-strict inequalities could be sampled if the cones differ on the sampled feature, equivalently either equality or non-strict inequality could be selected if the cones do not differ on that dimension — e.g., {#1,#2} $>(x_1,x_2,\text{size})$, or {#3,#4} $\geq(x_1,x_2,\text{size})$.

3. **Extend.** With probability $\frac{1}{2}$ go to Step 4, otherwise sample uniformly one of the following expansions and repeat. For statements with two bound variables Step 3 is performed for x_1, then again for x_2:

   (a) **Conjunction.** A cone is sampled from the subset picked out by the statement thus far and one of its features sampled — e.g., {#1} $\wedge(=(x_1,\text{green},\text{color}),\geq(x_1,\text{medium},\text{size}))$. Again, inequalities are sampled only if they increase the true set size relative to equality — e.g., "$\wedge(\leq(x_1,3,\text{xposition}),\geq(x_1,\text{medium},\text{size}))$", which picks out more objects than "$\wedge(=(x_1,3,\text{xposition}),\geq(x_1,\text{medium},\text{size}))$".

   (b) **Disjunction.** An additional feature-value pair is selected uniformly from *either* unselected values of the current feature, *or* from a different feature — e.g., $\vee(=(x_1,\text{color},\text{red}),=(x_1,\text{color},\text{blue}))$ or $\vee(=(x_1,\text{color},\text{blue}),\geq(x_1,\text{size},2))$. This step is skipped if the statement is already true of all the cones in the scene.

4. **Quantify.** Given the contained statement, select true quantifier(s):

   (a) For statements involving a single bound variable (i.e., those inspired by a single cone in Step 1) the possible quantifiers simply depend on the number of the cones in the scene for which the statement holds. The quantifier is selected uniformly from the existential $\exists(\_)$ or numerical "exactly $N^{=}(\_,J)$", "at least $N^{\geq}(\_,J)$", or "at most $N^{\leq}(\_,J)$" and J is set to match number of cones for which the statement is true. If it it is true for all cones, the universal quantifier $\forall(\_)$ may also be selected.

   (b) Statements involving two bound variables in lambda calculus have two nested quantifier statements each selected as in (a). The inner statement quantifying x_2 is selected first based on truth value of the expression while taking x_1 to refer to the cone observed in '1.'. The truth of the selected inner quantified statement is then assessed for all cones to select the outer quantifier — e.g., {#3,#4} "$\wedge(=(x_2,\text{green},\text{color}),\leq(x_1,x_2,\text{size}))$" might become "$\forall(\lambda x_1\colon \exists(\lambda x_2\colon \wedge(=(x_2,\text{green},\text{color}),\leq(x_1,x_2,\text{size})),\mathcal{X}),\mathcal{X})$". The inner quantifier $\exists$ is selected because three of the four cones are green {#1, #2, #4}, and the outer statement is selected because all cones are less than or equal in size to a green cone.

Note that a procedure like the one laid out above is, in principle, capable of generating any rule generated by the PCFG in Table 1, but will only do so when exposed to a positive

[3] Numbers prepended with # refer to the labels on the cones in the example observation in Figure 2b.

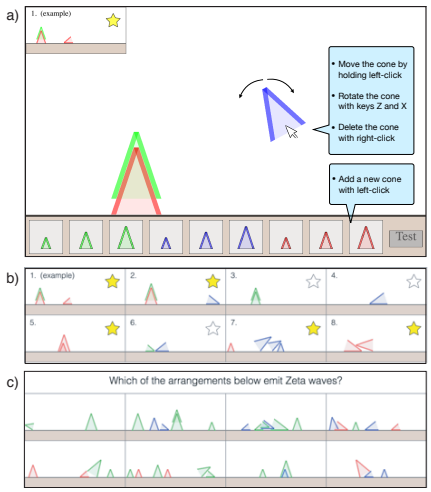

Figure 3: a) Active learning phase. b) A sequence of 8 tests, the first is provided as an example, and subsequent tests are constructed by the learner using the interface in (a). c) Prediction phase: Participants select 1-7 of the scenes by clicking on them.

observation that is actually consistent with that rule, and will do so more often when the observation is inconsistent with as many other rules as possible (i.e., a minimal positive example). Basing hypotheses on instances should improve the quality of the effective sample of rules that the learner generates.

One way to think of the IDG procedure is as an inversion of a PCFG. As illustrated by the blue text in the examples in Figure 2, while the PCFG starts at the outside and works in, the IDG starts from the central content and works outward out to a quantified statement, ensuring at each step that this final statement is true of the scene.

We now report on an experiment designed to test human concept learning using it to assess whether people behave more in line with PCFG or IDG hypothesis generation,

Experiment

Our pilot experiment places participants in the learning environment outlined in the introduction in which they must learn about concepts by testing encountered and self generated scenes and then make predictions about whether new scenes are consistent with the concept.

Methods

Participants Thirty people (10 female, age 36.8 ± 11.5 [M±SD]) took part and were paid $7 per hour plus a bonus of up to $4 ($2.6 \pm 0.3$) depending on the accuracy of their predictions. The task took 34.2 ± 16.4 minutes.

Materials The experiment was programmed in javascript using a port of the Box2D physics game engine and was run online using Psiturk (Gureckis et al., 2016). The scenes were displayed and constructed using an interactive 800 by 500 pixel window (see Figure 3 and try the task here: `https://github.com/neilbramley/discovery`).

Test rules We chose a range of test rules expressible in first order logic and lambda calculus, that varied in their complexity and the features they relate to (see Table 2).

Table 2: Rules Tested in Experiment

	General	Specific	Initial Example
1.	*Pair–value:*	There's a red $\exists(\lambda x_1: \ =(x_1, \text{red}, \text{color}), \mathcal{X})$	
2.	*Match:*	They're all the same size $\forall(\lambda x_1: \forall(\lambda x_2: \ =(x_1, x_2, \text{size}), \mathcal{X}), \mathcal{X})$	
3.	*Negation:*	Nothing is upright $\forall(\lambda x_1: \ \neg(=(x_1, \text{upright}, \text{orientation})), \mathcal{X})$	
4.	*Numerosity:*	There is exactly 1 blue $\text{exactly}(\lambda x_1: \ =(x_1, \text{blue}, \text{color}), 1, \mathcal{X})$	
5.	*Conjunct:*	There's something blue and small $\exists(\lambda x_1: \ \wedge(=(x_1, \text{blue}, \text{color}), =(x_1, 1, \text{size}), \mathcal{X})$	
6.	*Disjunct:*	All are blue or small $\forall(\lambda x_1: \ \vee(=(x_1, \text{blue}, \text{color}), =(x_1, 1, \text{size}), \mathcal{X})$	
7.	*Relative property:*	A red is the largest piece $\exists(\lambda x_1: \forall(\lambda x_2: \wedge(=(x_1, \text{red}, \text{color}), > (x_1, x_2, \text{size})), \mathcal{X}), \mathcal{X})$	
8.	*General relation:*	Some pieces are touching $\exists(\lambda x_1: \exists(\lambda x_2: \Gamma(x_1, x_2, \text{contact}), \mathcal{X}), \mathcal{X})$	
9.	*Specific relation:*	A blue touches a red $\exists(\lambda x_1: \exists(\lambda x_2: \wedge(\wedge(=(x_1, \text{blue}, \text{color}), = (x_2, \text{red}, \text{color})), \Gamma(x_1, x_2, \text{contact})), \mathcal{X}), \mathcal{X})$	
10.	*Complex:*	Some pieces are stacked $\exists(\lambda x_1: \exists(\lambda x_2: \wedge(\wedge(\wedge(\wedge(= (x_1, \text{upright}, \text{orientation}), = (x_1, \text{yes}, \text{grounded})), = (x_2, \text{upright}, \text{orientation})), = (x_2, \text{no}, \text{grounded})), = (x_1, x_2, \text{xpos}), \Gamma(x_1, x_2, \text{contact})), \mathcal{X}), \mathcal{X})$	

Procedure Participants were given an "alien planet" cover story in which their task was to establish why some sets of cones emit strange new forms of radiation. Participants were orientated toward the features relevant to the experiment though examples of five forms of radiation that had already been discovered (as in Figure 1). They then completed 10 test problems corresponding to the rules in Table 2 in random order.

Each problem had a learning phase, and a prediction phase. In the *learning phase*, participants were shown an initial scene that did follow the rule (shown in Table 2) and were then allowed to perform 7 additional tests on scenes of their own construction. Participants created their own scenes using a construction interface (illustrated in Figure 3a) in which they could add cones using a set of buttons, remove them with a right-click on them and move them around by holding left-click on them. They would then click a "test" button and the interface would calculate whether the scene followed the rule, and if so displayed emitted radiation, visualized as gold stars rising from the scene. A record of the current problems' sequence of tests and their outcomes was displayed at the top of the window with gold stars indicating which of these scenes had emitted radiation (see Figure 3b). In the *prediction phase*, participants were asked to make predictions about which of 8 new cases would emit radiation by selecting at least 1 and less than 8 of the scenes (Figure 3c). The prediction scenes were selected from a large set generated by sequential addi-

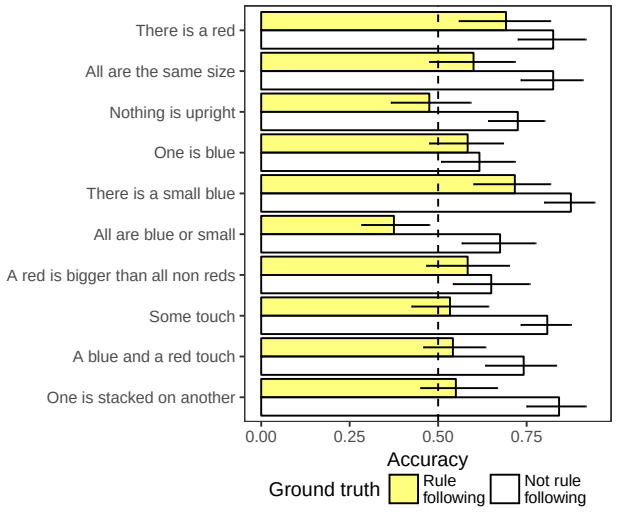

Figure 4: Generalization accuracy by question type and ground truth. Errorbars show bootstrapped SE.

tion of random cones using the construction interface. Four rule-following and four non-rule-following scenes were selected at random for each problem and shared across participants, but their on-screen position was randomized independently. Finally, participants provided a written description of their best guess about the hidden rule which we do not analyze here.

Results and discussion

We first compare participants predictions across different rules before fitting our PCFG and IDG learning models.

Participants correctly predicted the labels of $66 \pm 21\%$ (M$\pm$SD) of the 8 test scenes, correctly labeling negative instances more often than positive instances ($76 \pm 27\%$ compared to $57 \pm 32\%$ $t(29) = 4.4, p < .001$, Figure 4). Accuracy differed between rules as shown by repeated measures analysis $F(9,261) = 5.5, \eta_g^2 = .14, p < .001$. While substantially below 100%, participants predictions' did correspond with the ground truth were significantly above the chance level of 50% (with a corrected p<.005 level) for all rules except 3. "Nothing is upright", 4. "There is exactly 1 blue", and 6. "All are blue or small".

The lowest performance was on disjunctive rule 6. "All are blue or small" replicating the classic finding that people struggle to learn disjunctive rules (Shepard, Hovland, & Jenkins, 1961).

Quantitative comparison of PCFG and IDG

We now compare participants predictions to a PCFG and a IDG learning model as well as a random Baseline model. We first drew 50,000 samples from our PCFG with uniform production weights. We then used participants generalizations to fit the production weights to the empirical frequencies in the human data. To do this, we gave every production an initial weight of 1. We then took the set of m rules consistent with each participant's rule-following scenes in the first two phases (active learning and test phase, Figure 3a and b), and added $1/m$ to each of the production weights before renormalizing these weights within each group of productions for the same symbol (i.e., within each row of Table 1). We then resampled 10,000 candidate rules using the fitted hypothesis space for all 10 problems. The full set of fitted parameters is available on `https://github.com/neilbramley/discovery`. Most are not substantially different from uniform but there was evidence of preferential use of the color property in participants' rules over the other features: [*color=.32*, *size=.14*, *orientation=.10*, *grounded = .10*, *x-position*=.11, *y-position*=.11]. We hypothesized that the IDG would benefit from the scenes' natural statistics by sampling uniformly across cones so did not attempt to fit its stochastic elements but gave it the same feature selection probabilities derived from the PCFG above. We then used the IDG model to generate a separate set of 10,000 rules spread equally across the positive trials experienced by each participant on each trial. For each prediction phase, we generated posteriors for both models by subsampling only those hypotheses consistent with all eight outcomes the participant experienced during the learning phase.

We then gave each model a single decision noise parameter τ controlling a soft maximization over the averaged predictions of the surviving rules. For both models, the probability $P(s)_{pt}$ that scene s is rule-following on trial t based on data generated by participant p is thus given by

$$P(s)_{pt} = \frac{1}{|R||\mathbf{d}_{pt}|} \sum_{r \in R||\mathbf{d}_{pt}} r(\mathcal{X}_s) \tag{1}$$

where $R||\mathbf{d}_{pt}$ is the subset of the initial set of rules consistent with data $\mathbf{d}_{pt}$ and $r(\mathcal{X}_s)$ is the application of the rule to cones $\mathcal{X}_s$, outputting 1 if they follow the rule or 0 otherwise. The probability participant p selects s is thus a soft maximization over these probabilities

$$P(\text{choose}_{pt} = s) = \frac{e^{P(s)_{pt}/\tau}}{e^{P(s)_{pt}/\tau} + e^{(1-P(s)_{pt})/\tau}} \tag{2}$$

where $\tau \to 0$ indicates hard maximization over $P(s)$, while $\tau \to \infty$ indicates random responding. We fit τ for both models using maximum likelihood and compared against a Baseline model that treats each prediction as a coin flip. The results, detailed in Table 3, show that IDG provides a quantitatively better fit to participants patterns than PCFG, both in terms of fitting more participants individually, and in terms of having the lowest overall BIC value. However a small majority of individual participants were better fit by the Baseline random selection model. The PCFG was slightly more accurate than IDG at at matching the ground truth (72% compared to 70%) but the moderate accuracy of both models even with large samples highlights the difficulty of the learning task.

Since we hypothesized that IDG would be more effective for bounded learners who are limited in the number of samples they can store and evaluate, we also compared the models restricting them to smaller number of initial samples, contrasting performance with 1, 10, 100 and 1000 samples. For IDG these samples were evenly spread across the positive

Table 3: Model Fits to Participants' Predictions

Model	LogL	BIC	τ	ES	N/30	Acc
Baseline	-1663	3327	∞		17	0.500
PCFG	-1594	3195	1.54	352	3	0.722
IDG	**-1539**	**3085**	**1.01**	**610**	**10**	**0.702**

Note: ES = effective samples: the average number of hypotheses remaining at the generalization phase. N/30 = number of particiants best fit.

cases encountered by the learner on each problem. Figure 5 shows the results of these tests. IDG retains a larger number of effective hypotheses by the end of the task than PCFG for an equivalent initial sample size, and this results in more accurate predictions in the task for 1, 10, and 100 samples. IDG described participants better (i.e., smaller BIC) than PCFG and Baseline for 10, 100, and 1000 samples, while for a single sample, the parameter free Baseline fit best. Participant accuracy was closest to the 1000 sample variants of the models.

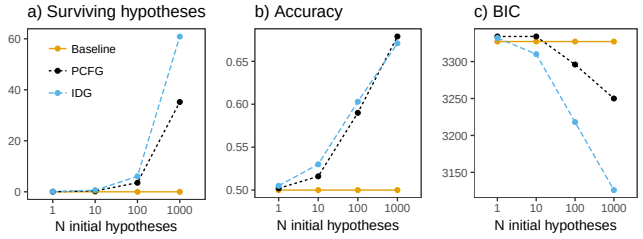

Figure 5: Small sample performance of PCFG and IDG.

While IDG beat the PCFG model, neither provided a tight fit to the participants predictions within the rule-following and non-rule-following test scenes. One possibility is that this variance stems from perceptual ambiguities in evaluating the hypotheses on some of these scenes.

General Discussion

This paper contrasted two proposals about how people generate hypotheses about concepts. We found support for the idea that people learn in a partly instance driven way as described by our Instance Driven Generator.

Space constraints prohibited analysis of participants testing choices, but we expect these to provide an additional window on concept learning. To the extent that positive examples inspire hypotheses, we expect learners to benefit from "minimal" positive examples over more complex cases that maximize information. The idea that participants' hypotheses mirror positive examples could help explain positive testing biases (Klayman & Ha, 1987). We note also that we do not solve the problem of identifying relevant features.

From instance driven generation to instance driven adaptation

While generating hypotheses inspired by observations is a good start for a bounded concept learner, most precise hypotheses still do not have a long life expectancy in the face of new data. Thus a natural partner to the IDG approach is to try to *adapt* hypotheses to account for new data as it arrives (Bramley, Dayan, Griffiths, & Lagnado, 2017; Dasgupta, Schulz, & Gershman, 2016; Trabasso & Bower, 1975). For example, Lewis et al. (2014) explored an error driven hypothesis adaptation mechanism that augments a hypothesis with a new disjunction whenever it is ruled out due to a false negative — e.g., "there is a green cone *or a red cone*" — and with a new conjunction whenever it is ruled out through a false positive — e.g., "there is a *large* green cone" (see also Nosofsky, Palmeri, & McKinley, 1994, for a related rule+exception approach). These are examples of a class of rule learning algorithm called sequential covering (Michalski, 1969) in which a model is expanded with a new component to cover every new encountered case. Going forward we plan to augment our the IDG proposal with rules for adapting hypotheses by considering small edits to their logical form as data arrives. In this way we hope to better capture the computational tricks by which human learners maintain a useful set of candidate hypotheses during learning, and so learn successfully even in infinite hypothesis spaces.

Acknowledgments NB was supported by the Moore Sloan foundation, JT by ONR grant (N00014-13-1-0333), FX by an NSF grant (#1640816), and TG by NSF grant (BCS-1255538) and a John McDonnell Foundation Scholar Award.

References

Bramley, N. R., Dayan, P., Griffiths, T. L., & Lagnado, D. A. (2017). Formalizing Neurath's ship: Approximate algorithms for online causal learning. *Psychological Review*, *124*(3), 301–338.
Church, A. (1932). A set of postulates for the foundation of logic. *Annals of mathematics*, 346–366.
Dasgupta, I., Schulz, E., & Gershman, S. J. (2016). Where do hypotheses come from? *Center for Brains, Minds and Machines (preprint)*.
Feldman, J. (2000). Minimization of Boolean complexity in human concept learning. *Nature*, *407*(6804), 630.
Fodor, J. A. (1987). *Psychosemantics: The problem of meaning in the philosophy of mind* (Vol. 2). MIT press.
Ginsburg, S. (1966). *The mathematical theory of context free languages*. McGraw-Hill Book Company.
Gold, E. M., et al. (1967). Language identification in the limit. *Information and control*, *10*(5), 447–474.
Goodman, N. D., Tenenbaum, J. B., Feldman, J., & Griffiths, T. L. (2008). A rational analysis of rule-based concept learning. *Cognitive Science*, *32*(1), 108–154.
Goodman, N. D., Tenenbaum, J. B., & Gerstenberg, T. (2015). Concepts in a probabilistic language of thought. In E. Margolis & S. Lawrence (Eds.), *The conceptual mind: New directions in the study of concepts* (pp. 623–653). MIT Press.
Gureckis, T. M., Martin, J., McDonnell, J., Rich, A. S., Markant, D., Coenen, A., … Chan, P. (2016). Psiturk: An open-source framework for conducting replicable behavioral experiments online. *Behavior research methods*, *48*(3), 829–842.
Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, *57*(1), 97–109.
Klayman, J., & Ha, Y.-W. (1987). Confirmation, disconfirmation, and information in hypothesis testing. *Psychological Review*, *94*(2), 211.
Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, *350*(6266), 1332–1338.
Lewis, O., Perez, S., & Tenenbaum, J. (2014). Error-driven stochastic search for theories and concepts. In *Proceedings of the 36th Annual Meeting of the Cognitive Science Society* (Vol. 36). Cognitive Science Society Austin, TX.
Michalski, R. S. (1969). On the quasi-minimal solution of the general covering problem. In (Vol. A3, pp. 125–128).
Nosofsky, R. M., Palmeri, T. J., & McKinley, S. C. (1994). Rule-plus-exception model of classification learning. *Psychological Review*, *101*(1), 53.
Piantadosi, S. T., Tenenbaum, J. B., & Goodman, N. D. (2016). The logical primitives of thought: Empirical foundations for compositional cognitive models. *Psychological Review*, *123*(4), 392.
Schulz, L. (2012). Finding new facts; thinking new thoughts. In *Advances in Child Development and Behavior* (Vol. 43, pp. 269–294). Elsevier.
Shepard, R. N., Hovland, C. I., & Jenkins, H. M. (1961). Learning and memorization of classifications. *Psychological monographs: General and applied*, *75*(13), 1.
Trabasso, T., & Bower, G. H. (1975). *Attention in learning: Theory and research*. Krieger.