AFRL-AFOSR-VA-TR-2019-0280

Bayesian Program Learning and Concept Induction

Josh Tenenbaum
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

09/19/2019
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Materiel Command

DISTRIBUTION A: Distribution approved for public release.
AF Office Of Scientific Research (AFOSR)/ RTA2
Arlington, Virginia 22203
Air Force Research Laboratory

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
21-09-2019

2. REPORT TYPE
Final Performance

3. DATES COVERED *(From - To)*
15 Dec 2015 to 14 Dec 2018

4. TITLE AND SUBTITLE
Bayesian Program Learning and Concept Induction

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-16-1-0012

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Josh Tenenbaum

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
77 MASSACHUSETTS AVE
CAMBRIDGE, MA 02139-4301 US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AF Office of Scientific Research
875 N. Randolph St. Room 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR RTA2

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-VA-TR-2019-0280

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
A long-standing dream in computing has been to build machines that learn like a child Turing (1950) that grow into all the kinds and forms of knowledge that human adults do, starting from much less. At a minimum, any such learning system must be able to acquire many dierent kinds of expertise. Every child becomes an expert in natural language, motor control, intuitive physics and social interaction, and many will grow into adults with specialized expertise in cooking, calculus, tennis, drawing pictures, or writing software. Despite great advances, artificial intelligence (AI) is still far from acquiring human-like expertise in any of these domains let alone all of them, as a person can.

15. SUBJECT TERMS
Bayesian Induction, Machine Learning, Concept Formulation, A Priori Knowledge, Probabilistic Inference

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			RIECKEN, RICHARD
Unclassified	Unclassified	Unclassified	UU		19b. TELEPHONE NUMBER (Include area code) 703-941-1100

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Final Report for AFOSR award FA9550-16-1-0012

September 12, 2019

A long-standing dream in computing has been to build machines that learn like a child Turing (1950) — that grow into all the kinds and forms of knowledge that human adults do, starting from much less. At a minimum, any such learning system must be able to acquire many different kinds of expertise. Every child becomes an expert in natural language, motor control, intuitive physics and social interaction, and many will grow into adults with specialized expertise in cooking, calculus, tennis, drawing pictures, or writing software. Despite great advances, artificial intelligence (AI) is still far from acquiring human-like expertise in any of these domains — let alone all of them, as a person can.

This gap points to a fundamental feature of human learning. When people become experts in a domain, they learn not only specific tasks but also more general learning and thinking abilities: how to solve new problems faster and better than novices can, expressing their solutions in a far more compact, more explanatory form with a conceptual vocabulary that novices do not have access to and would hardly understand. Today's machine learning systems, in contrast, are built to solve specific tasks without this more abstract level of expertise. AI systems learn to play challenging games at superhuman levels, but cannot explain the strategies they come to, or transfer what they learn to related games or variants, as a human expert can. Language models may generate convincing English text within the styles they are trained on, but they do not learn to analyze the abstract structures that generalize across different languages, as a linguist does, nor learn to learn the meanings of new words more quickly, as a child does when learning language (Smith et al., 2002; Kemp et al., 2007).

In this report, we summarize our projects in Bayesian concept learning from both AI and cognitive science perspectives that were supported by the AFOSR award FA9550-16-1-0012. In Sections 1-5, we summarize projects on program induction which resulted in the following publications: **(Ellis et al., 2018a)**, **(Ellis et al., 2019)**, **(Ellis et al., 2018b)**, **(Ellis et al., 2016)**, and **(Nye et al., 2019)**. In Section 6, we summarize a project on deep hierarchical Bayesian modeling applied to one-shot learning of handwritten characters from new alphabets: **(Hewitt et al., 2018)**. In Sections 7-9, we summarize projects on computational models of cognitive processes for learning concepts: **(Rule et al., 2015)**, **(Rule et al., 2018)**, **(Rule et al., 2019)**.

1 DreamCoder (Ellis et al., 2018a)

We present a computational model called DreamCoder that takes a step towards machines that can grow into genuine domain experts. Our work is inspired by cognitive science suggesting that expertise takes two complementary forms. First, experts learn explicit, declarative concepts that are abstract yet finely-tuned to their domain. Artists learns concepts like arcs, spirals, symmetries, and perspectives; physicists learn concepts like inner products, vector fields, and conservation laws; and architects learn concepts like arches, supports, and bridges. Second, experts acquire implicit, procedural skill in deploying those concepts quickly to solve new problems. Compared to novices, experts more faithfully classify problems based on the "deep structure" of their solutions Chi et al. (1981); Chi and VanLehn (2012), seeing their underlying commonalities and intuiting which compositions of concepts are likely to solve a task even before searching for a solution. These two forms of expertise bootstrap each other: As learners build increasingly rich conceptual systems, more problems have easy solutions, but the challenge of producing the best solution to a problem only becomes harder, because the solution space expands with each new concept. We aim to build AI that builds expertise as humans do, learning the right explicit concepts together with the implicit skills to use those concepts effectively as knowledge grows. Figure 1 illustrates our proposed system and Figure 2 shows shows samples from our system applied to the LOGO drawing domain.

2 Write, Execute, Assess: Program Synthesis with a REPL (Ellis et al., 2019)

We present a neural program synthesis approach integrating components which write, execute, and assess code to navigate the search space of possible programs. We equip the search process with an interpreter or a read-eval-print-loop (REPL), which immediately executes partially written programs, exposing their semantics. The REPL addresses a basic challenge of program synthesis: tiny changes in syntax can lead to huge changes in semantics. We train a pair of models, a policy that proposes the new piece of code to write, and a value function that assesses the prospects of the code written so-far. At test time we can combine these models with a Sequential Monte Carlo algorithm. We apply our approach to two domains: synthesizing text editing programs and inferring 2D and 3D graphics programs. In Figure 3, we show examples of programs synthesized by our system.

3 Learning to Infer Graphics Programs from Hand-Drawn Images (Ellis et al., 2018b)

We introduce a model that learns to convert simple hand drawings into graphics programs written in a subset of LaTeX. The model combines techniques from

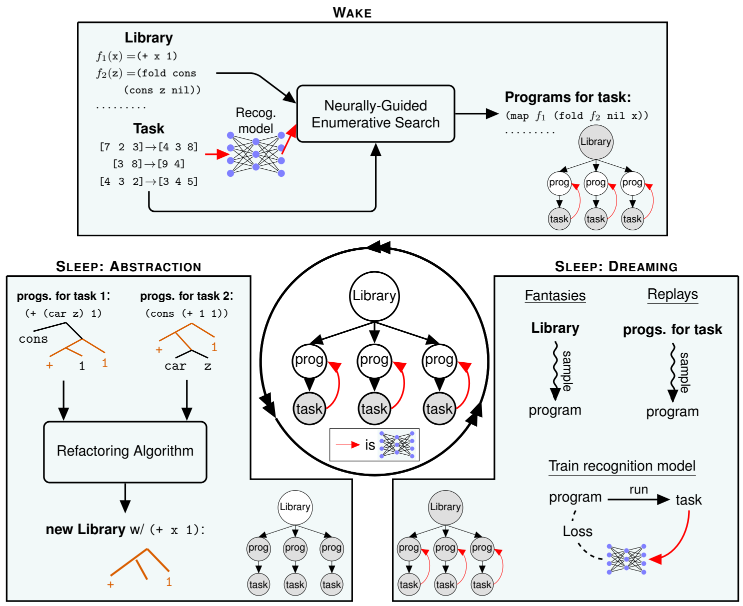

Figure 1: The Bayesian model underlying our model and its basic algorithmic cycle. **Middle:** DreamCoder as a graphical model. Agent observes programming tasks (e.g., input/outputs for list processing or images for graphics programs), which it explains with latent programs, while jointly inferring a latent library capturing cross-program regularities. A neural network, called the *recognition model* (red arrows) is trained to quickly infer programs with high posterior probability. **Top**: Wake phase infers programs while holding the library and recognition model fixed. **Left**: Sleep (Abstraction) phase updates library while holding the programs fixed by refactoring programs found during waking and abstracting out common components (highlighted in orange). **Right**: Sleep (Dreaming) phase trains recognition model to predict approximate posterior over programs conditioned on task. Trained on 'Fantasies' (programs sampled from library) & 'Replays' (programs found during waking).

deep learning and program synthesis. We learn a convolutional neural network that proposes plausible drawing primitives that explain an image. These drawing primitives are a specification (spec) of what the graphics program needs to draw. We learn a model that uses program synthesis techniques to recover a graphics program from that spec. These programs have constructs like variable bindings, iterative loops, or simple kinds of conditionals. With a graphics program in hand, we can correct errors made by the deep network and extrapolate drawings. Figure 4 shows the kinds of extrapolations enabled by our system.

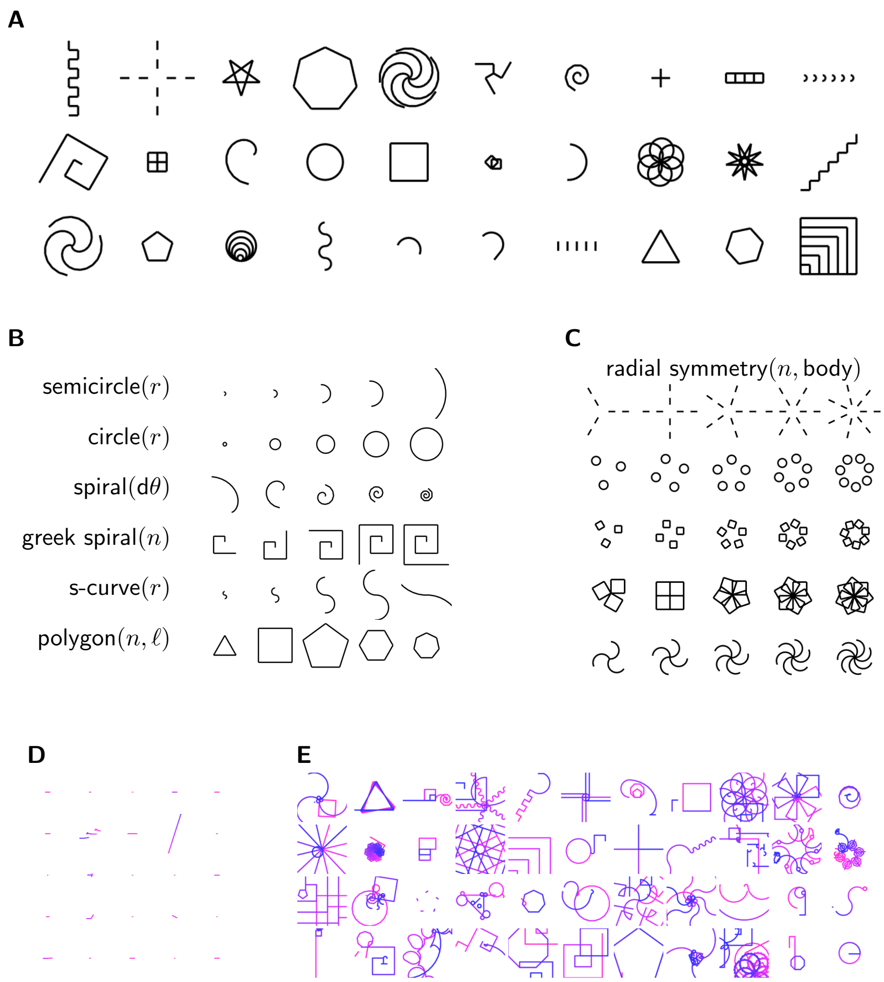

Figure 2: **(A):** 30 (out of 160) LOGO graphics tasks. Agent writes a program controlling a 'pen' that draws the target picture. **(B-C):** Example learned library routines. Agent learns interpretable parametric routines for drawing families of curves (**B**) as well as primitives that take entire programs as input (**C**). Each row of images on the left is the same code executed with different parameters. Each image on the right is the same code executed with different parameters and with a different subprogram provided as input. **(D-E):** dreams, or sampled programs, from library before (**D**) and after learning (**E**, most interesting dreams selected from 5 different runs), showing how dreams become more complex with acquired expertise. Blue: where the agent started drawing. Pink: where the agent ended drawing.

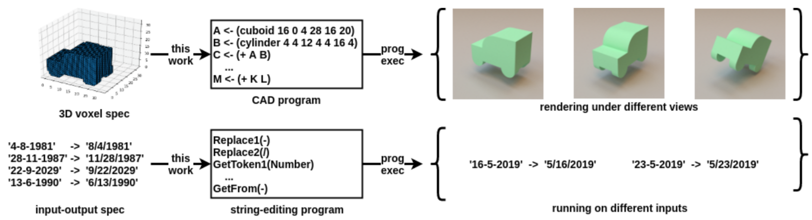

Figure 3: Examples of programs synthesized by our system. Top, graphics program from voxel specification. Bottom, string editing program from input-output specification.

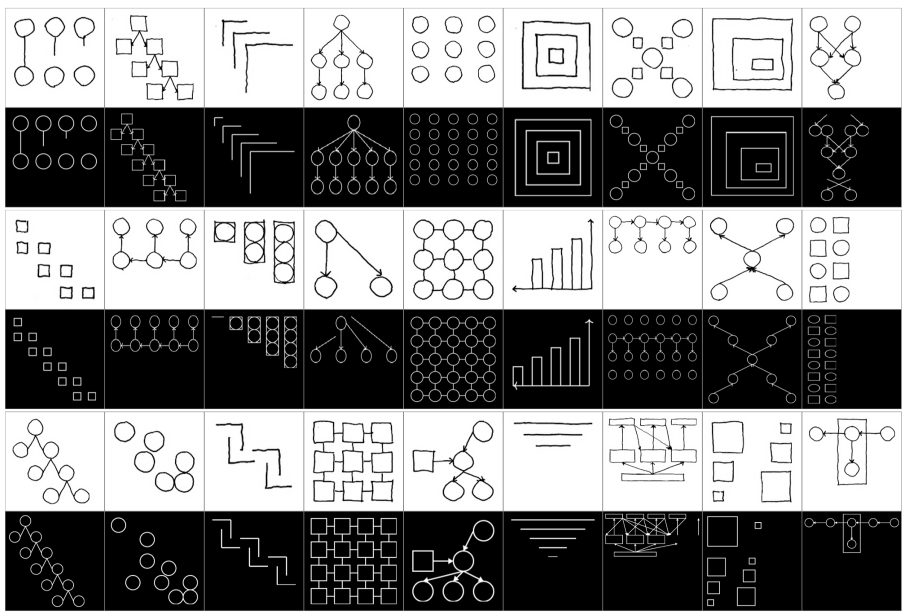

Figure 4: Top, white: drawings. Bottom, black: extrapolations automatically produced by our system.

4 Sampling for Bayesian Program Learning (Ellis et al., 2016)

Towards learning programs from data, we introduce the problem of sampling programs from posterior distributions conditioned on that data. Within this setting, we propose an algorithm that uses a symbolic solver to efficiently sample programs. The proposal combines constraint-based program synthesis with sampling via random parity constraints. We give theoretical guarantees on how well the samples approximate the true posterior, and have empirical results showing the algorithm is efficient in practice, evaluating our approach on 22 program learning problems in the domains of text editing and computer-aided programming.

5 Learning to Infer Program Sketches (Nye et al., 2019)

Our goal is to build systems which write code automatically from the kinds of specifications humans can most easily provide, such as examples and natural language instruction. The key idea of this work is that a flexible combination of pattern recognition and explicit reasoning can be used to solve these complex programming problems. We propose a method for dynamically integrating these types of information. Our novel intermediate representation and training algorithm allow a program synthesis system to learn, without direct supervision, when to rely on pattern recognition and when to perform symbolic search. Our model matches the memorization and generalization performance of neural synthesis and symbolic search, respectively, and achieves state-of-the-art performance on a dataset of simple English description-to-code programming problems. Figure 5 shows an overview of our model.

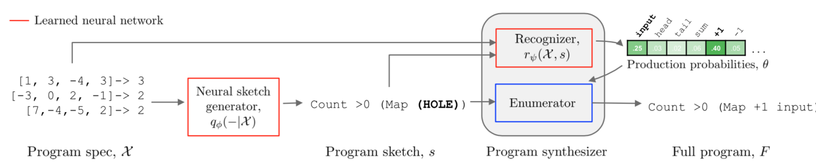

Figure 5: Schematic overview of our model. A program spec (in the form of examples) is fed into a sketch generator, which outputs a distribution over sketches. In our experiments, the neural sketch generator is parametrized by a seq-to-seq recurrent neural network with attention. The program sketch is given to a program synthesizer, which searches for full programs which satisfy the spec. Our enumerative synthesizer is guided by a learned recognizer, which is conditioned on the spec and the sketch and predicts the likelihood of using each program token to fill in the sketch.

6 The Variational Homoencoder (Hewitt et al., 2018)

Hierarchical Bayesian methods can unify many related tasks (e.g. k-shot classification, conditional and unconditional generation) as inference within a single generative model. However, when this generative model is expressed as a powerful neural network such as a PixelCNN, we show that existing learning techniques typically fail to effectively use latent variables. To address this, we develop a modification of the Variational Autoencoder in which encoded observations are decoded to new elements from the same class. This technique, which we call a *Variational Homoencoder* (VHE), produces a hierarchical latent variable model which better utilises latent variables. We use the VHE framework to learn a hierarchical PixelCNN on the Omniglot dataset, which outperforms all existing models on test set likelihood and achieves strong performance on one-shot generation and classification tasks. We additionally validate the VHE on natural images from the YouTube Faces database. Finally, we develop extensions of the model that apply to richer dataset structures such as factorial and hierarchical categories. Figure 6 shows examples of one-shot learning enabled by our model.

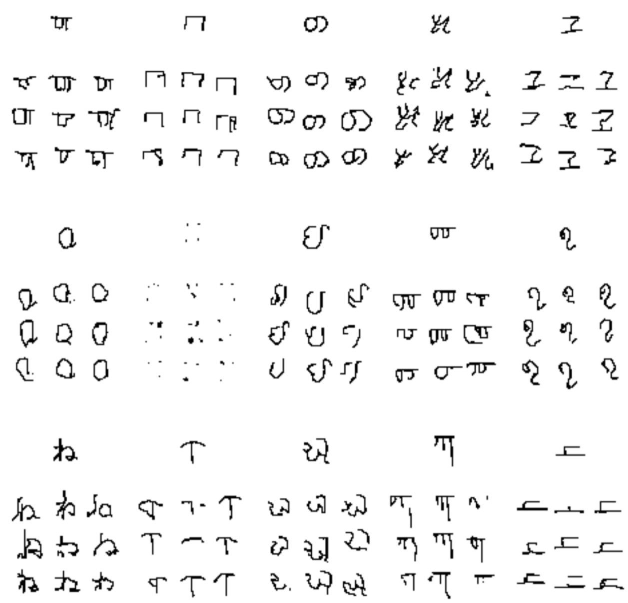

Figure 6: One-shot same-class samples generated by our model. Cue images were sampled from previously unseen classes.

7 Representing and Learning a Large System of Number Concepts with Latent Predicate Networks (Rule et al., 2015)

Conventional models of exemplar or rule-based concept learning tend to focus on the acquisition of one concept at a time. They often underemphasize the fact that we learn many concepts as part of large systems rather than as isolated individuals. In such cases, the challenge of learning is not so much in providing stand-alone definitions, but in describing the richly structured relations between concepts. The natural numbers are one of the first such abstract conceptual systems children learn, serving as a serious case study in concept representation and acquisition (Carey, 2009; Fuson, 1988; Gallistel & Gelman, 2005). Even so, models of natural number learning focused on single-concept acquisition have largely ignored two challenges related to natural number's status as a system of concepts: 1) there is an unbounded set of exact number concepts, each with distinct semantic content; and 2) people can reason flexibly about any of these concepts (even fictitious ones like *eighteen-gazillion*). To succeed, models must instead learn the structure of the entire infinite set of number concepts, focusing on how relationships between numbers support reference and generalization. Here, we suggest that the latent predicate network (LPN) – a probabilistic context-sensitive grammar formalism – facilitates tractable learning and reasoning for natural number concepts (Dechter, Rule & Tenenbaum, 2015). We show how to express several key numerical relationships in our framework, and how a Bayesian learning algorithm for LPNs can model key phenomena observed in children learning to count. These results suggest that LPNs might serve as a computational mechanism by which children learn abstract numerical knowledge from utterances about number. Figure 7 shows a comparison between data from real child learning and our model.

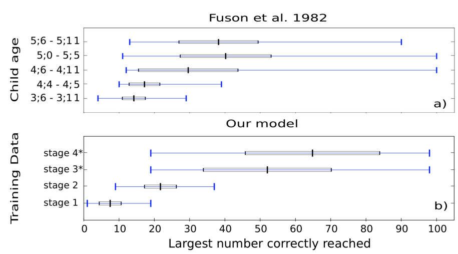

Figure 7: Our model compared with children's counting data a) Data from Fuson et al. (1982). The x-axis shows the highest number correctly reached when children were asked to count starting at "one." Boxes correspond to the standard deviation, central bands to the means, and whiskers to the range. b) Model performance, averaged over ten runs at four stages of increasing data quantity.

8 Learning list concepts through program induction (Rule et al., 2018)

Humans master complex systems of interrelated concepts like mathematics and natural language. Previous work suggests learning these systems relies on iteratively and directly revising a language-like conceptual representation. We introduce and assess a novel concept learning paradigm called Martha's Magical

Machines that captures complex relationships between concepts. We model human concept learning in this paradigm as a search in the space of term rewriting systems, previously developed as an abstract model of computation. Our model accurately predicts that participants learn some transformations more easily than others and that they learn harder concepts more easily using a bootstrapping curriculum focused on their compositional parts. Our results suggest that term rewriting systems may be a useful model of human conceptual representations.

9 Learning a novel rule-based conceptual system (Rule et al., 2019)

Humans have developed complex rule-based systems to explain and exploit the world around them. When a learner has already mastered a system's core dynamics—identifying its primitives and their interrelations—further learning can be effectively modeled as discovering useful compositions of these primitives. It nevertheless remains unclear how the dynamics themselves might initially be acquired. Composing primitives is no longer a viable strategy, as the primitives themselves are what must be explained. To explore this problem, we introduce and assess a novel concept learning paradigm in which participants use a two-alternative forced-choice task to learn an unfamiliar rule-based conceptual system: the MUI system (Hofstadter, 1980). We show that participants reliably learn this system given a few dozen examples of the system's rules, leaving open the mechanism by which novel conceptual systems are acquired but providing a useful paradigm for further study.

References

Alan M Turing. Computing machinery and intelligence. *Mind*, 1950.

Linda B Smith, Susan S Jones, Barbara Landau, Lisa Gershkoff-Stowe, and Larissa Samuelson. Object name learning provides on-the-job training for attention. *Psychological science*, 13(1):13–19, 2002.

Charles Kemp, Amy Perfors, and Joshua B Tenenbaum. Learning overhypotheses with hierarchical bayesian models. *Developmental science*, 10(3):307–321, 2007.

Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum. Learning libraries of subroutines for neurally–guided bayesian program induction. In *Advances in Neural Information Processing Systems*, pages 7805–7815, 2018a.

Kevin Ellis, Maxwell Nye, Yewen Pu, Felix Sosa, Josh Tenenbaum, and Armando Solar-Lezama. Write, execute, assess: Program synthesis with a repl. *arXiv preprint arXiv:1906.04604*, 2019.

Kevin Ellis, Daniel Ritchie, Armando Solar-Lezama, and Josh Tenenbaum. Learning to infer graphics programs from hand-drawn images. In *Advances in neural information processing systems*, pages 6059–6068, 2018b.

Kevin Ellis, Armando Solar-Lezama, and Josh Tenenbaum. Sampling for bayesian program learning. In *Advances in Neural Information Processing Systems*, pages 1297–1305, 2016.

Maxwell Nye, Luke Hewitt, Joshua Tenenbaum, and Armando Solar-Lezama. Learning to infer program sketches. *arXiv preprint arXiv:1902.06349*, 2019.

Luke B Hewitt, Maxwell I Nye, Andreea Gane, Tommi Jaakkola, and Joshua B Tenenbaum. The variational homoencoder: Learning to learn high capacity generative models from few examples. In *Uncertainty in Artificial Intelligence*, 2018.

Joshua Rule, Eyal Dechter, and Joshua B Tenenbaum. Representing and learning a large system of number concepts with latent predicate networks. In *CogSci*, pages 2051–2056, 2015.

Joshua Rule, Eric Schulz, Steven T Piantadosi, and Joshua B Tenenbaum. Learning list concepts through program induction. *BioRxiv*, page 321505, 2018.

Joshua Rule, Steven T Piantadosi, and Joshua B Tenenbaum. Learning a novel rule-based conceptual system. 2019.

Michelene TH Chi, Robert Glaser, and Ernest Rees. Expertise in problem solving. Technical report, PITTSBURGH UNIV PA LEARNING RESEARCH AND DEVELOPMENT CENTER, 1981.

Michelene TH Chi and Kurt A VanLehn. Seeing deep structure from the interactions of surface features. *Educational Psychologist*, 2012.

Karen C Fuson, John Richards, and Diane J Briars. The acquisition and elaboration of the number word sequence. In *Children's logical and mathematical cognition*, pages 33–92. Springer, 1982.