Leveraging Unstructured Statistical Knowledge in a Probabilistic Language of Thought

Alexander K. Lew[1], **Michael Henry Tessler**[2], **Vikash K. Mansinghka**[2], **and Joshua B. Tenenbaum**[2]
{alexlew, tessler, vkm, jbt}@mit.edu
[1] Department of Electrical Engineering and Computer Science, MIT [2] Department of Brain and Cognitive Sciences, MIT

Abstract

One hallmark of human reasoning is that we can bring to bear a diverse web of common-sense knowledge in any situation. The vastness of our knowledge poses a challenge for the practical implementation of reasoning systems as well as for our cognitive theories – how do people represent their common-sense knowledge? On the one hand, our best models of sophisticated reasoning are top-down, making use primarily of symbolically-encoded knowledge. On the other, much of our understanding of the statistical properties of our environment may arise in a bottom-up fashion, for example through associationist learning mechanisms. Indeed, recent advances in AI have enabled the development of billion-parameter language models that can scour for patterns in gigabytes of text from the web, picking up a surprising amount of common-sense knowledge along the way—but they fail to learn the structure of coherent reasoning. We propose combining these approaches, by em- bedding language-model-backed primitives into a state-of-the-art probabilistic programming language (PPL). On two open-ended reasoning tasks, we show that our PPL models with neural knowledge components characterize the distribution of human responses more accurately than the neural language models alone, raising interesting questions about how people might use language as an interface to common-sense knowledge, and suggesting that building probabilistic models with neural language-model components may be a promising approach for more human-like AI.

Keywords: probabilistic language of thought; language models; neurosymbolic reasoning; common sense

Introduction

Imagine a heavy package is delivered to your neighbor's door. You start to think: *What might be in the box?* One of the most astonishing features of human knowledge is how flexibly it can be deployed. Even in such an unconstrained task, you can imagine information that could be relevant to constrain your imagination of possibilities: *did your neighbor just move in?*, *are they having groceries delivered?*, *is that furry animal they have been walking around with a new pet?*

Bayesian reasoning provides a principled avenue to modeling this kind of *conditional imagination*: imagined worlds can be seen as samples from a Bayesian posterior, which combines a broad prior on probable worlds with lightly informative evidence that eliminates some possibilities while lending credence to others. One challenge for this approach, however, is specifying the model: it is difficult to write down probability distributions over possible worlds that capture our vast web of common-sense knowledge. The *probabilistic language of thought* hypothesis offers some guidance, by giving a concrete representational system by which people might make their knowledge available for probabilistic reasoning (Goodman, Tenenbaum, & Gerstenberg, 2014). According to the theory, our knowledge of the world is organized into *concepts* that we combine in language-like ways. The content of a concept is a function or subroutine in a probabilistic programming language; when faced with a new situation, we draw on a rich library of these concept building blocks to compose an appropriate model of the situation on the fly, much as a programmer might code up a script in Python. The resulting model—a program in the probabilistic language of thought—encodes a probability distribution over world-states that is sufficiently precise to reason in combinatorial ways. The value of such a probabilistic representational system enters when faced with a novel reasoning problem: evidence can be used to update prior beliefs and reason to complex conclusions. Indeed, the inferences derived from simple probabilistic programs applied to novel reasoning problems (e.g., evaluating the strength of individuals in a tug-of-war tournaments) closely match those of human intuitions (Gerstenberg & Goodman, 2012; Gerstenberg & Tenenbaum, 2017).

Another way we might we acquire and deploy knowledge to imagine possible worlds in the context of evidence is bottom-up associationist learning. Many cognitive tasks require knowledge about the statistical properties of our environment (Rogers & McClelland, 2004), which need not be richly structured in an explicit manner. For example, language could serve as a source of associationist knowledge: when words co-occur, it tells us that the things they pick out are related (Griffiths, Steyvers, & Tenenbaum, 2007). Perhaps our imagination of what is in the neighbor's delivery box is best modeled not as probabilistic reasoning, but as a direct appeal to learned associations between properties of our observations (that something was delivered in a box, and that it was heavy) and possible answers. Indeed, recent advances in AI highlight the richness that is latent in the statistics of the world and especially language; powerful neural language models like BERT and GPT-2 (Devlin, Chang, Lee, & Toutanova, 2018; Radford et al., 2019) can not only generate long strings of fluent text, but also answer an impressive variety of questions, like who directed *The Hateful Eight* and what a rabbit typically eats. But the models' knowledge is locked away in the uninterpretable weights of a neural network, and it is not clear how to leverage it for open-ended reasoning tasks (McCoy, Pavlick, & Linzen, 2019; Dasgupta, Guo, Stuhlmüller, Gershman, & Goodman, 2018).

In this paper, we explore a middle road between richly structured program-like representations and vast but unstructured statistical knowledge as a candidate model for human reasoning and a potential approach for more human-like artificial intelligence. We embed primitives for querying associationist sources of knowledge, such as neural language models, into a probabilistic programming language, enabling structured, probabilistic reasoning of unstructured knowledge. We implement our computational framework as a library on top of a state-of-the-art probabilistic programming language called Gen (Cusumano-Towner, Saad, Lew, & Mansinghka, 2019).[1] Using the library, Gen probabilistic programs can query a statistical language model as a proxy for common-sense knowledge that would be difficult to encode manually, for example, to sample an object that might be found in a kitchen or generate a reasonable price for a laptop. These queries gently elicit associations and individually, do not tax the language model heavily. But by embedding the queries to the neural language model in the context of a structured probabilistic program, the model can answer a variety of open-ended reasoning questions featuring diverse patterns of evidence. We test the model's ability to exhibit human-like reasoning patterns in two open-ended verbal reasoning domains and compare our hybrid model to state-of-the-art neural language models adapted to perform the same tasks. We find preliminary evidence that our structured, representational system builds upon the knowledge latent in neural language models to exhibit more human-like reasoning, suggesting a particular hypothesis for how structured and unstructured modeling approaches should be integrated.

Computational Framework

Our approach is rooted in the Probabilistic Language of Thought hypothesis, according to which people build generative models for various tasks compositionally, using as building blocks a library of *concepts*, or probabilistic functions. Each function $f_i(x_1,\dots,x_{n_i})$ encodes an input-dependent distribution $p_{f_i}(y_1,\dots,y_{m_i} \mid x_1,\dots,x_{n_i})$ over a set of random variables $y_{1\dots m_i}$. New functions can be built using all the tools one expects to see in a programming language: sequential composition, branching, looping, and recursion.

Reasoning with probabilistic programs

For example, consider the models in Figure 1. At the top is a program called **roll_dice**, which chooses a die (either 4-, 6-, 12-, or 20-sided), and then rolls it several times. The program syntax is similar to that of Python, with assignment statements and for loops, but with the addition of random primitives that draw samples from uniform or Poisson distributions. One way to read the program is as a simulator, which could be run many times to simulate many die-rolling episodes. But if we implement **roll_dice** in a *probabilistic programming language*, like Church, WebPPL, or

[1] Code: https://github.com/alex-lew/luskplot-cogsci-2020

Program 1: A Structured Model of Die-Rolling

```
function roll_dice()
 num_sides = uniform([4, 6, 12, 20])
 num_rolls = poisson(3)

 for i=1:num_rolls
  rolls[i] = uniform(1:num_sides)
 end

 total = sum(rolls)
 return num_sides, num_rolls, rolls, total
end

# Run the model conditioned on observations
query(roll_dice, observe(:num_rolls => 2,
                          :total => 7))
query(roll_dice, observe(:rolls[1] => 3))
```

Program 2: A Structured Model of Shopping with Unstructured Statistical Knowledge

```
function go_shopping()
 store = noun("I went to the [?] store.")
 num_items = poisson(3)

 for i=1:num_items
  items[i]  = noun("I bought this [?]
                at the $(store) store.")
  prices[i] = associated_quantity(
               items[i], "dollars")
 end

 total = sum(prices)
 return store, items, prices, total
end

query(go_shopping, observe(:store => "grocery"))
query(go_shopping, observe(:total => 500))
```

An Unstructured Approach to Shopping

```
# Sampling a shopping list (p(items))
gpt2_next_para("I bought the following
               things today:")

# Inferring a store (p(store | price))
gpt2_next_word("I spent 500 dollars today,
               at the following shop:")
```

Figure 1: Three models. **roll_dice:** A probabilistic program that models a scenario in which a die is chosen (4-, 6-, 12-, or 20-sided) and rolled a random number of times. The rolls are then summed. Although the structure of this model is similar to many real-world scenarios—for example, choosing a store to shop at, buying some unpredetermined number of items, and summing their prices before you pay—it is harder to model those scenarios because we lack an exhaustive model of stores, the items in them, and prices. **GPT-2:** Questions about shopping can be tackled directly with language models, which learn about the world by processing gigabytes of text. But the structure in the model is lost, and the language model may not respect the constraints we wish to impose. **go_shopping:** This work combines the two approaches, by adding language-model-backed primitives like **noun**, for sampling a noun to fill a hole in a sentence, and **associated_quantity**, for sampling a number associated with a noun.

Gen (Goodman, Mansinghka, Roy, Bonawitz, & Tenenbaum, 2012; Goodman & Stuhlmüller, 2014; Cusumano-Towner et al., 2019), we can take advantage of a dual interpretation of the program, as precisely specifying a joint probability distribution over a collection of random variables:

$$p(sides, n, roll_{1\dots n}, total) = \frac{3^n e^{-3}}{4(n!)} \prod_i \frac{1}{sides} \mathbb{I}\left[total = \Sigma_i roll_i\right].$$

Using this representation, we can apply probabilistic reasoning to solve complex queries, such as: *If the sum of the rolls was 31, how many sides did the die likely have?* Or: *If a six-sided die was rolled one or more times to produce a sum of 8, how many times was it likely rolled?* The **query** statements in Figure 1 show how this is done: we can pass in a list of observations on which to *condition* the model, and the system will sample (perhaps approximately) from the *posterior distribution* of any unobserved variables. For example, the line `query(roll_dice, observe(:num_rolls => 2, :total => 7))` will sample from $p(sides, roll_{1\dots 2} \mid n = 2, total = 7)$. In this case, the knowledge that the sum of two rolls was 7 will mean that a plurality of the posterior samples we draw will have $sides = 6$, reflecting the fact that two 6-sided die rolls are more likely to sum to 7 than are two 4-, 12-, or 20-sided die rolls.

A challenge: common-sense knowledge

The die-rolling scenario is contrived, but its structure is not: there are many parts of our lives that might be well-modeled by a similar program. For example, shopping involves choosing a store (just as we had to choose a die), buying several items at the store (rolling the die), and summing the prices of the items to obtain a total (as with the die). Just as in the die-rolling example, we can imagine posing many queries about shopping: *If the final amount on the credit card statement is $500, where did my partner likely go shopping?* Or: *If the total at the shoe store is $50, how many pairs of shoes were likely purchased?* The probabilistic language of thought hypothesis gives an appealing answer to the problem of how people reason so flexibly about situations like this. The challenge is that this scenario is hard to encode as a probabilistic program. Die-rolling is possible to model with an impoverished library of concepts, including **uniform**, **poisson**, and little else. To build a model of going shopping, we would need to incorporate a lot of knowledge about the world, including what kinds of stores exist, what sorts of things are sold in each store, and how much different items tend to cost.

Neural language models

Modern deep learning models provide a different kind of an answer to such queries. For example, one could invoke the neural language model GPT-2 (Radford et al., 2019) with prompts that encode the query of interest in English (Figure 1). Because GPT-2 has been exposed to billions of words from the web during training, we might expect it to have acquired some of the "common-sense" knowledge required for this sort of reasoning task. But because there is so little structure in what the model produces, it often fails to respect the constraints we wish to impose; when asked to generate a shopping list, it may not output a list at all, and when conditioning on the total amount spent, GPT-2 does not appear particularly sensitive to the specific dollar amount we use.

A middle ground

In this work, we propose to combine the strengths of both approaches. We add primitives to the Gen probabilistic programming language for querying single values from large models trained on web-scale corpora of text. The **noun** function chooses a single word to fill in the blank in a user-specified prompt, using the XLNet neural network model (Yang et al., 2019), a masked language model trained to predict words given bidirectional context. The **associated_quantity** function samples a number in a given unit (here, *dollars*) associated with a specified noun. It is backed by a database collected by Elazar, Mahabal, Ramachandran, Bedrax-Weiss, and Roth (2019), who scraped the web for mentions of various noun phrases in conjunction with quantities in different units. Our **associated_quantity** primitive accepts as input a noun and a unit, and fits a log-normal mixture model to all mentions of that noun available in the database. Using these primitives, we can modify the **roll_dice** program to obtain the middle program of Figure 1, **go_shopping**. Structurally, it is the same as the die-rolling model, except instead of sampling integers, it samples words. Running the program forward generates multiple queries to language models, first to choose a store, then to generate each purchased item from the store, then to calculate the prices of each item. But since it is written in a probabilistic programming language, we can also treat the program as defining a distribution we can manipulate: denoting by s the store, n the number of items, r_1 and r_2 the two template strings we use to prompt XLNet, and $(o_{1\dots n}, c_{1\dots n})$ the individual purchases and their costs, we have

$$p(s, n, o, c) = \frac{3^n e^{-3}}{n!} f_{XL}(s \mid r_1) \prod_i f_{XL}(o_i \mid r_2(s)) q_{\$}(c_i \mid o_i),$$

where $f_{XL}(s \mid r)$ is the probability that the XLNet neural model assigns to a word s given a prompt string r, and $q_{\$}(c \mid o)$ is the probability density function, evaluated at a number c, of the log-normal mixture model fit to the database of dollar-valued quantities mentioned in relation to object o. We can use inference in this structured model (the **query** statements in the figure) to answer many different queries, e.g., sampling likely shopping lists conditioned on the location being a grocery store, or on the total price being $500.

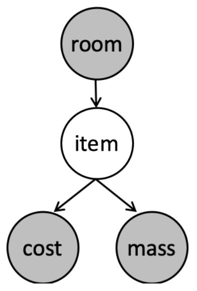
Application to Verbal Reasoning

We use this computational framework to write probabilistic program models for two families of verbal reasoning problems which we test with a behavioral experiment.

Items in the home. We consider simple verbal reasoning problems like the following:

Items in the Home

"I've been told to look for something in the *room*, and I've been told it costs \$*cost* and weighs *mass* grams. What could it be?"



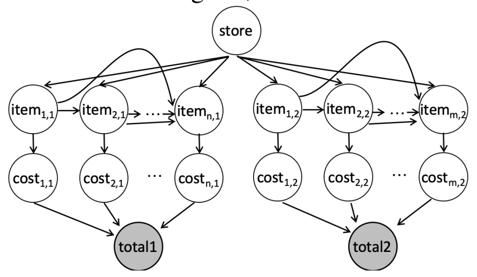
```
function items_in_the_home()
 room = noun("There are many rooms
             in a home, such as
             the [?].")
 item = noun("There are many objects
             in the $(room), such as
             this [?].")

 cost = associated_quantity(
        item, "dollars")
 mass = associated_quantity(
        item, "grams")
 return room, item, cost, mass
end
```

```
query(items_in_the_home,
      observe(:room => "kitchen",
              :cost => 50))
```

Samples:
kettle,
saucepan,
mixer

Friends Going Shopping

"My friend and I went to the same store. I bought *n* items for a total of $total_1$, and my friend bought *m* items for a total of $total_2$. What store did we go to, and what did we each buy?"



```
query(friends_shopping(n=2, m=1),
      observe(:total_1 => 30,
              :total_2 => 150))
```

Sample:
corner (store),
sandwich & cake,
bicycle

Figure 2: Models for two classes of verbal reasoning problem. The edges in the graphical models correspond to invocations of our new probabilistic programming primitives, **`noun`** and **`associated_quantity`**.

I've been told to look for something in the **kitchen**. I've been told that it costs **\$20**, but nothing else. What could it be?

In different versions of the question, we might be looking for an item in a room other than the kitchen, and we might know the *mass* of the object in question, in addition to or instead of the price. To handle this family of prompts, we write a probabilistic program (Figure 2, left) that implements a generative model of all four variables of interest. It begins by sampling a room and an item in it, using **`noun`**, then samples a cost and mass for the item, using **`associated_quantity`**. At query time, we constrain the room (in Figure 2, to equal *kitchen*) as well as the cost or mass (or both). Querying the model with *cost* set to \$50 generates samples of objects that might plausibly be found in a kitchen and that cost \$50: a kettle, a saucepan, a mixer, and so on.

Friends going shopping. We now turn to a more complex family of questions. Here is a representative example:

My friend and I found some receipts, but can't remember what we bought. I know we both went shopping at the **same** store. I spent **\$30** and got **2** things, and my friend spent **\$150** and got **1** thing. What kind of store could we have gone to, and what do you think we each bought?

In different instances, we might vary whether the friends go to the same or different stores; how much each spent; and on how many items. We might also reveal the identity of one or more of the purchased items. The probabilistic program that models this family of scenarios is slightly more complex, and we do not show the full code here, but the graphical model induced by the program is depicted in Figure 2's right-hand side (for the case where the friends go to the same store). Conditioning on the total prices, we can sample reasonable values for the unknown variables. For the problem above, one sample from the model is that we went to a corner store, where I bought a cake and sandwich and my friend bought a bicycle. But given the open-ended nature of the task, different runs will give different outcomes.

Experiment

Our experiment is designed to characterize human responses to problems of the form described above. These problems elicit very open-ended responses; our probabilistic approach allows us to predict and characterize the *distribution* of responses that people give. Indeed, the lack of a single correct answer is what motivated our application of a Bayesian model: we expect to see a given response more often if it is more plausible *a priori* or if it explains the evidence particularly well, the two factors that Bayes' rule trades off.

Participants We recruited 113 participants from Amazon's Mechanical Turk. Participants were restricted to those with U.S. IP addresses and who had at least a 95% work approval rating. The experiment took on average 20 minutes and participants were compensated \$3.00 for their work.

Materials Our materials set was composed of simple and complex stimuli corresponding to the *Items in the Home* and *Friends Going Shopping* verbal reasoning problems described in the previous section. For *Items in the Home*, each stimulus is parameterized by a location (one of *bedroom*, *bathroom*, *kitchen*, *living room*, or *basement*) and either a cost in dollars, a weight in pounds, or both. These were randomly generated each trial, by first generating a band (low, medium, or high), and then one of a fixed set of quantities within that band. Low prices ranged from \$3 to \$20; medium from \$40 to \$100, and high from \$250 to \$1000. Low masses ranged from 1.5 to 5 pounds; medium from 10 to 20 pounds, and high from 40 to 90 pounds. For *Friends Going Shopping*, we developed 12 stimuli, organized into 6 minimal pairs that differed in only one respect (whether the friends went to the same store; which items are revealed ahead of time to have

Table 1: Qualitative Results: Minimal Pairs.

Cheap Socks	$20	$2000
Store	clothing (62%)	department (35%)
	department (17%)	clothing (33%)
entropy	*1.38*	*1.98*
Friend's items	shirt (36%)	tv (18%)
	pants (17%)	suit (16%)
entropy	*1.85*	*2.90*
Magazine	Different store	Same store
Store 1	book (65%)	gun (27%)
	convenience (16%)	department (22%)
entropy	*1.64*	*2.27*
Other items	book (50%)	book (15%)
	gum (5%)	bullets (12%)
entropy	*2.09*	*2.89*

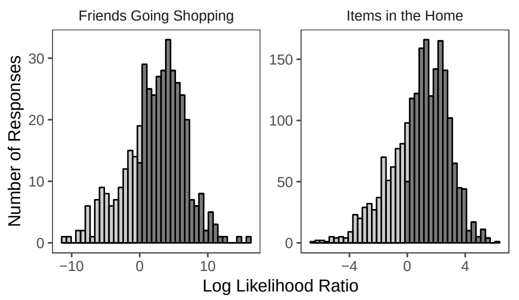


Figure 3: Log ratios (our model to the baseline) of posterior probabilities of human responses, on two verbal reasoning tasks. In both tasks, more than 70% of respondents' answers are better explained by our model than by the baseline.

been purchased; and the total cost of one friend's items).

Procedure Participants were introduced to two cartoon characters that needed their help identifying objects. Each participant completed 24 trials, consisting of 18 *Items in the Home* stimuli and 6 *Friends Going Shopping* stimuli. On each trial, the participants were presented with a prompt of the form described in the previous section posed as a request by the cartoon character. For *Items in the Home*, participants were required to enter a single-word answer in a text box. For *Friends Going Shopping*, participants entered multiple text responses for the type(s) of store the friends went to and the items each friend purchased. After collection, we removed any responses not in the XLNet language model's vocabulary.

Qualitative Results

To illustrate qualitative behavior, we describe responses to minimal pairs of *Friends Going Shopping* stimuli.

Cheap socks. In the *cheap socks* prompts, two friends go to the same store, where one of them spends $10 on socks and one other (unspecified) item. Participants are told that the second friend has spent either $20 or $2000; Table 1 shows how participant responses depend on which stimulus they see.

When the second friend is known to have spent $20, participants have little uncertainty about what kind of store the friends visited: 62% say it was a clothing store, and most participants name *shirt* or *pants* as the $20 item. This certainty is reflected in the low entropy of the empirical response distributions. But when the second friend spends $2000, the percentage of participants who name *clothing* as the type of store drops to 33%, eclipsed by the more general *department* store. The distribution over items purchased changes significantly as well, placing significant mass on *tv* and *suit*. Overall, uncertainty increases, which makes sense: people are rightly confused to hear that one friend bought cheap socks for under $10 at the same store where the other spent two grand.

Magazine. In the *magazine* prompts, participants are told that one friend has purchased a magazine and another item, for $30, whereas the other has purchased a shotgun. In one of the prompts, the friends went to the same store; in the other, different stores. When the stores are different, participants can reason individually about each, and often say that the magazine was purchased at a book store or convenience store, where a book or pack of gum might also have been purchased. But higher-entropy answer distributions result when the magazine and shotgun come from the same store. Some participants appear to realize that *magazine* can have a gun-related meaning, and name 'gun' as the store type.

Model-Based Analysis

To evaluate if probabilistic structure helps better capture the distribution of human responses, we compare the probability $p(response \mid prompt)$ our model assigns to each response to that given by task-specific neural baselines (Figure 3).

Neural baselines. For each stimulus, we design a prompt designed to elicit an answer from a neural language model directly, without additional probabilistic reasoning. The neural model's output distribution on words is then compared with the posterior distribution under our model.

For *Items in the Home*, we compare our model to a baseline that uses a single call to XLNet, with prompts of the form, "There are many objects in the *[location]*, including this *[price if it exists, e.g. $20]*, *[weight if it exists, e.g. 3-pound]* [?]." This allows the network to see the evidence that we expect should constrain the hypothesis space (the weight or price). We use a mask with a period token afterward to encourage the model to place high probability on good one-word answers (just as we instructed human respondents).

For *Friends going Shopping*, we compared to a baseline model that generated responses sequentially using GPT-2, on the sentence, "I went to a store, in particular the [?] store, and spent $*[total price 1]* on the following *[number of items 1]* purchases: *[any known items here]*, this [?], . . . , and this [?]. My friend went to the same store *[or: 'a different store, the [?] store']*, and spent $*[total price 2]* on the following *[number of items 2]* purchases: *[any known items here]*, this [?], . . . , and this [?]." That is, GPT-2 was given each word of this prompt sequentially, and whenever a '[?]' was encountered, GPT-2 was queried and we evaluated the probability that it assigned to the human response We use an autoregressive model rather than a masked language model for this task, because a masked language model cannot jointly fill all the blanks coherently in one pass.

Items in the home. For the "items in the home" stimuli, we collected 2,592 responses, of which 2,077 were in-vocabulary for our model. Each response is a single word – an object – chosen based on a room in the home and certain properties (price and/or weight). We evaluate the posterior probability that our model assigns to each human response, which requires computing the normalizing constant $p(\mathit{price}, \mathit{mass} \mid \mathit{room})$; we estimate this constant by summing over the 2000 *a priori* most probable objects for each room. We found our model assigned higher probability than the baseline model to 71% of the human responses. The mean log ratio of our model probability to the baseline was $0.89 = \ln(2.4)$, i.e. our model appears to narrow the space of good answers by a modest factor (2-3x reduction). In many of the cases where our model performed poorly, human participants seemed not to take some of the given information into account. For example, one respondent answered *potatoes* when asked to name a \$10, 60-pound object in a bedroom. A weakness of our model is that correlation between price and weight factors through object identity, which can lead to counter-intuitive behavior. For example, two participants named "shampoo" as an item in a bathroom weighing 15 pounds, at the price points \$8 and \$90 respectively. Our model assigns low probability to both, because 15-pound shampoo is rare. But it assigns especially low probability to the \$90 response, believing it to be an unlikely price for shampoo. However, the shampoo might have been purchased in bulk, explaining both the high price and high weight. Another failure mode for the model is polysemous words, like "tablet": many participants listed tablet as a four- or five-pound object, but our model assumes they weigh just a few grams, based on *tablet*'s pharmaceutical meaning.

Friends going shopping. For the "friends going shopping" stimuli, we collected 648 responses (418 were in-vocabulary). Each response consists of one-word store types, as well as comma-separated lists of purchased items for each. We estimated normalizing constants for each prompt via importance sampling with 200 particles[2]. We find our model assigns higher probability than the baseline to 73% of responses, and the mean log ratio between the probabilities is 2.36. The model exhibited similar failure modes as described above. For example, the model does not capture that the same store charges similar prices for similar objects; conditioned on an object, it treats the prices as independent.

Discussion

The ability to reason open-endedly about possible worlds in light of evidence raises questions both for cognitive science (how is our knowledge represented to enable such reasoning?) and the engineering of human-like AI (how can we build models that deploy common-sense knowledge?). The Probabilistic Language of Thought is an intuitively appealing and quantitatively compelling framework (Goodman et al., 2014), but has little to say about how to architect the rich library of probabilistic concepts necessary for the symbolic representation of everyday scenarios.

In this work, we present a technique that takes a first step toward investigating these questions. Incorporating language models into probabilistic programs enables the exploration of a modeling space in which some knowledge is encoded symbolically (the structure of the probabilistic program), and some is implicit in queries to language model-backed primitives. In experiments, we show that on two open-ended reasoning tasks, our model—a point in this modeling space—better characterizes the distribution of human answers than alternatives that lack the probabilistic structure.

From a knowledge engineering point of view, neural language models are a convenient primitive because they are trained once in a task-general way, but can be applied to approximate a more accurate model in many different contexts. For example, people could build a rich, detailed causal model of the prices of objects; but for the simple task of imagining a \$10-object that might be found in a kitchen, it may be cheaper to rely on a generic associationist modeling component. This sort of "amortized modeling,'' in which generically useful modeling components are built at great expense, but can be applied very cheaply, may be an interesting avenue for future research, even if the interface onto these generic modeling components is not taken to be natural language.

Our work leaves many parts of the story unwritten. For example, each model contains hand-crafted *prompts* that serve as a somewhat brittle interface between a structured probabilistic program and associative resources like XLNet. We cannot simply sample an item likely to be found in a kitchen, but instead must sample a *word* to fill in the blank in a sentence, e.g. "There are many objects in a kitchen, including this [?]." Future work must develop a more satisfying account of this interface, and from an AI engineering perspective, a less brittle engineering discipline for using these components.

Indeed, from a cognitive perspective, it is unclear whether language models should be used to model the way people access common-sense knowledge. On the one hand, we learn a lot about the world from what others tell us and some knowledge may be available only through linguistic associations; i.e. language is one way in which a community may divide the labor of categorizing the world (Putnam, 1975). On the other hand, a lot of common sense is present before we are walking and talking (Spelke & Kinzler, 2007) and if any knowledge is truly "common sense," people would rarely talk about it. Finally, common sense is not necessarily common in the sense that it is shared by everyone; we each learn different associations that are specific to our beliefs, values, and cultures. A single language model learned from billions of documents is a poor stand-in.

[2]We experimented with larger numbers of particles but observed little to no change in estimates.

Acknowledgments

This material is based upon work supported by the National Science Foundation SBE Postdoctoral Research Fellowship Grant No. 1911790 awarded to M.H.T. and Army Research Office MURI Grant No. W911NF-19-1-0057.

References

Cusumano-Towner, M. F., Saad, F. A., Lew, A. K., & Mansinghka, V. K. (2019). Gen: a general-purpose probabilistic programming system with programmable inference..

Dasgupta, I., Guo, D., Stuhlmüller, A., Gershman, S. J., & Goodman, N. D. (2018). Evaluating compositionality in sentence embeddings.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding.

Elazar, Y., Mahabal, A., Ramachandran, D., Bedrax-Weiss, T., & Roth, D. (2019). How large are lions? inducing distributions over quantitative attributes.

Gerstenberg, T., & Goodman, N. (2012). Ping pong in church: Productive use of concepts in human probabilistic inference. In *Proceedings of the annual meeting of the cognitive science society* (Vol. 34).

Gerstenberg, T., & Tenenbaum, J. B. (2017). Intuitive theories. *Oxford handbook of causal reasoning*, 515–548.

Goodman, N. D., Mansinghka, V., Roy, D. M., Bonawitz, K., & Tenenbaum, J. B. (2012). Church: a language for generative models.

Goodman, N. D., & Stuhlmüller, A. (2014). *The Design and Implementation of Probabilistic Programming Languages.* `http://dippl.org`. (Accessed: 2020-5-28)

Goodman, N. D., Tenenbaum, J. B., & Gerstenberg, T. (2014). Concepts in a probabilistic language of thought. In Margolis & Laurence (Eds.), *Concepts: New directions.* MIT Press.

Griffiths, T. L., Steyvers, M., & Tenenbaum, J. B. (2007). Topics in semantic representation. *Psychological review*, *114*(2), 211.

McCoy, R. T., Pavlick, E., & Linzen, T. (2019). Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference.

Putnam, H. (1975). The meaning of 'meaning'. *Philosophical papers*, *2*.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners.

Rogers, T. T., & McClelland, J. L. (2004). *Semantic cognition: A parallel distributed processing approach.* MIT press.

Spelke, E. S., & Kinzler, K. D. (2007). Core knowledge. *Developmental science*, *10*(1), 89–96.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems* (pp. 5754–5764).