Synthesizing Theories of Human Language with Bayesian Program Induction

John Smith,[1*] Jane Doe,[1] Joe Scientist[2]

[1]Department of Chemistry, University of Wherever,
An Unknown Address, Wherever, ST 00000, USA
[2]Another Unknown Address, Palookaville, ST 99999, USA

[*]To whom correspondence should be addressed; E-mail: jsmith@wherever.edu.

Introduction

An age-old aspiration within artificial intelligence research is to build a machine that automates the scientific process (*1, 2*), either by designing experiments (*3*) or synthesizing theories and models (*4*). However, gaining traction on the problem of theory induction requires a spectrum of tractable theory synthesis problems: while early work attempted to recapitulate classic discoveries like Kepler's laws (***?***), these results proved brittle and difficult to generalize. Despite these early works, theory induction has never attained the same level of success as machine vision or natural language processing. How can the artificial intelligence community get theory induction off the ground? This is an especially difficult question, because the current mainstream in machine intelligence focuses on learning to solve a qualitatively different class of problems (e.g., prediction tasks like classification and regression), whereas theory induction requires synthesizing human-understandable causal models of real-world phenomena (*5*), so that human scientists can understand and learn from the AI's outputs.

We propose theory induction research start with theories of human language, and scope out a collection of problems and techniques for a key module of natural language: *morphophonology*, the relationship between pronunciations of words and their meaning. Acquiring the morphophonology of a language involve solving a basic problem that confronts both linguists and children: given a collection of utterances, together with aspects of their meaning, what is the causal relationship between form and meaning? Morphophonology, and natural language more broadly, presents a tractable challenge to theory induction, for three reasons. First, every natural language is a different theory induction problem, so there is a wide spectrum of induction tasks to evaluate on. Second, linguistics as a field traditionally formalizes theories in computational terms, and so there is a pre-existing body of representations and algorithms for the AI researcher to draw upon. Third, we know that acquiring theories of language (e.g., grammars) is tractable, because children acquire language, and field linguists can be trained to build theories of grammar.

Our contribution to theory induction is an algorithm for synthesizing theories of natural language morphophonology. Like the linguist, the algorithm starts with a collection of utterances annotated with their meanings, and then constructs a causal, interpretable model explaining how those meanings gave rise to the utterances. An important aspect of a scientific theory is that it makes predictions; here, the theory predicts pronunciations of unobserved words conditioned on their meaning. We evaluate our algorithm on # data sets spanning # languages, automatically finding theories that can model a wide swath of a core component of human natural language.

Discovering Theories by Synthesizing Programs

We frame our approach as Bayesian Program Learning (BPL: see (*6*)), where the agent explains a set of utterance/meaning pairs $\{(u_n, m_n)\}_{n=1}^{N}$ by inferring a theory T, which we model

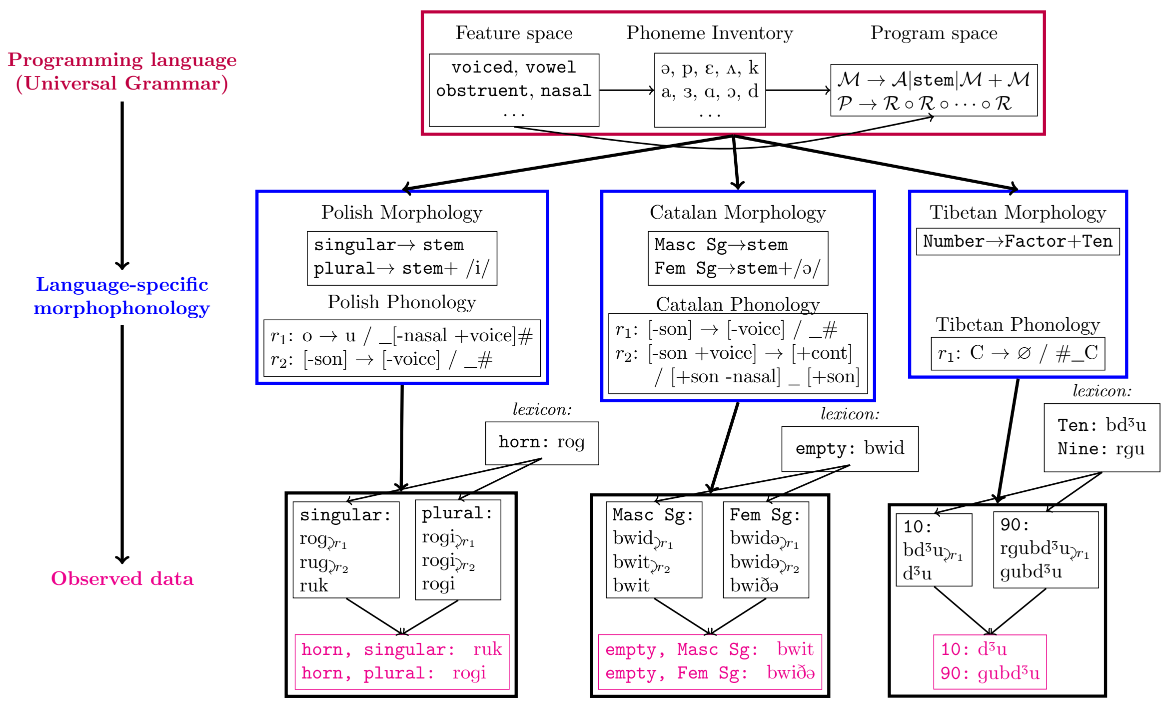
Programming language
(Universal Grammar)
Language-specific
morphophonology
Observed data
Feature space
voiced, vowel
obstruent, nasal
...
Phoneme Inventory
ə, p, ɛ, ʌ, k
a, ʒ, ɑ, ɔ, d
...
Program space
Polish Morphology
singular→ stem
plural→ stem+ /i/
Polish Phonology
Catalan Morphology
Masc Sg→stem
Fem Sg→stem+/ə/
Catalan Phonology
Tibetan Morphology
Number→Factor+Ten
Tibetan Phonology
lexicon:
horn: rog
lexicon:
empty: bwid
lexicon:
Ten: bdʒu
Nine: rgu
singular:
plural:
rogi
Masc Sg:
bwit
Fem Sg:
bwiðə
10:
dʒu
90:
gubdʒu
horn, singular: ruk
horn, plural: rogi
empty, Masc Sg: bwit
empty, Fem Sg: bwiðə
10: dʒu
90: gubdʒu

Figure 1:

Example data		Morphophonology
1	j̽ig	
4	š̽i	
5	ŋa	
9	ɡu	*Consonant cluster reduction:*
10	j̽u	C $\rightarrow \varnothing$ / #_C
11 $(= 10 + 1)$	j̽ugj̽ig	(*Upon encountering a consonant (C),*
14 $(= 10 + 4)$	j̽ubš̽i	*delete it ($\rightarrow \varnothing$) at the beginning of*
15 $(= 10 + 5)$	j̽uŋa	*a word (#_) if followed by another*
19 $(= 10 + 9)$	j̽urɡu	*consonant (_C)*)
40 $(= 4 + 10)$	š̽ibj̽u	
50 $(= 5 + 10)$	ŋabj̽u	
90 $(= 9 + 10)$	ɡubj̽u	

Table 1: Tibetan count system.

as a program. Formalizing grammars (theories) as generative programs has a long history in linguistics (*7*). Written as a probabilistic inference problem, the agent seeks the T maximizing $\left[\prod_{n=1}^{N} P(u_n, m_n|T)\right] P(T|UG)$, where UG is a "universal grammar" (**TIM**: what should I cite here?) constraining the space of allowed theories and imparting an inductive bias over the space of grammars. In this BPL setting we model UG as a prior distribution over theories (programs). To model the space of programs use Context-Sensitive Rewrites, a Turing-complete program representation, but restrict the rewrites in such a way as to make them equivalent to finite state transducers — this insight comes from the computational linguistics literature (*8*). For morphophonology, the relationship between meaning m_n and utterance u_n involves a latent *lexicon*, which is a mapping between lexemes and the pronunciation of their stem. For children, linguists, and our model, the lexicon must also be inferred (Fig. **??**).

Although this framing captures the problem a BPL theory inductor needs to solve, it offers no guidance on how to solve that problem: the space of all programs (theories) is infinitely large and sharply discontinuous, lacking the local smoothness that enables local optimization algorithms (e.g., gradient descent; MCMC) to succeed. We adopt a strategy currently unconventional to the AI community: constraint–based program synthesis, where the optimization problem is translated into a combinatorial constraint satisfaction problem and solved using a SMT solver (***?***).

Discussion

References

1. W. Paul, R. J. Solomonoff (1990).

2. P. Langley, *Scientific discovery: Computational explorations of the creative processes* (MIT Press, 1987).

3. A. Sparkes, *et al.*, *Automated Experimentation* **2**, 1 (2010).

4. M. Schmidt, H. Lipson, *science* **324**, 81 (2009).

5. J. Pearl, *Causality* (Cambridge university press, 2009).

6. B. M. Lake, R. Salakhutdinov, J. B. Tenenbaum, *Science* **350**, 1332 (2015).

7. N. Chomsky, M. Halle, *The sound pattern of English*, Studies in language (Harper & Row, 1968).

8. K. R. Beesley, L. Karttunen .