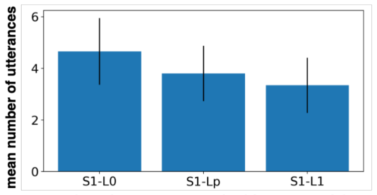
Thanks for the engaging comments. We will first address two common concerns, then address the reviewers individually.

Scalability to Larger Domains A universal concern is that this work will not scale to bigger synthesis domains. This is true. However, the focus of this paper is to define the computational problem and showing why solving it is valuable, which should in turn spur future research on efficient algorithms. For instance, one may imagine a compositional synthesis regime where the system communicates with the user to specify a component at a time (for instance, the layout domain maybe factorized in such a way that the user first specifies the shape patterns, then the color patterns). This way, rather than enumerating the entire space of programs, we enumerate one component at a time. Another solution is to amortize each component of the RSA model, L_0, S_1, L_1, with neural networks: A neural program synthesizer can serve as L_0. For natural language utterances, “Reasoning about Pragmatics with Neural Listeners and Speakers” (Andreas et. al.) gives a good construction for a pragmatic speaker S_1. For input-output based utterances, “Selecting Representative Examples for Program Synthesis” (Pu et. al.) gives a good construction for a pragmatic example-selector. Then, one can conceivably construct L_1 on top of the neurally approximated L_0 and S_1.

Comparison Against a Baseline that Uses a Prior Another common concern is the lack of a direct comparison against stronger baselines, specifically, one that leverages a good prior to disambiguate programs (a common strategy in previous works). Such priors can be obtained in 2 ways: by learning from real-world data, which is often expensive to obtain or unavailable, or by hand-crafting, which is appropriate for us. For our domain, a reasonable hand-crafted prior is one that prefers patterns with fewer numbers and kinds of symbols. We use L_p to denote the listener that first filters for consistent patterns, then ranks them using this prior. Using S_1 as the speaker, we compare the mean number of utterances required when L_0, L_p, L_1 are the listeners (see Fig). We note that L_1 and L_p have similar performances, despite L_1 being derived from the meaning matrix and the principles of pragmatic communication alone and without any hand-crafting. This is unsurprising, as S_1 and L_1 are *constructed* to be good “partners”. On the other hand, given a listener L_p that leverages an arbitrary prior, it is unclear how to obtain a good speaker “partner” for it other than directly optimizing one against it. We hope to add L_p to our user study to see whether end-users can find L_p as intuitive as L_1. Finally, a pragmatic listener can capture subtle aspects of communication *that no prior can model*: In the line game (Fig 2 of paper), if the utterance sequence is u_3, u_5, the most likely hypothesis according to L_1 is h_9, however, if the utterance reversed to u_5, u_3, the most likely hypothesis becomes h_0 instead. Thus, we have two valid hypothesis, h_0 and h_9, with their ordering flipped depending on the utterance. In contrast, a prior can only model a fixed global ordering over valid hypothesis.



number of utterances required for successful communication, error bars are 1 std.

R1 • “analysis ... consistent programs at each timestep”. There is not much difference, it really is the probability of L_1 that gives it the edge over L_0, will add to appendix. • “lines 103-106 ... which was small”. Sorry, by small we mean $p < 0.06$, so not quite $p < 0.05$. • “is this just the caching”. Yes, and also realizing parts of the summations are always 0. • “adds to the work of Wang et al”. The biggest difference would be their work is synthesis via translation, i.e. from NL instruction to PL execution. This work is synthesis from examples. Also, the number of candidate programs at each interaction “step” for Wang is ~ 30, for us it is ~ 3000. • “what is the runtime?”. We re-did the math more carefully, here is the result: Naively it is $|H|^2|U|k^2$, where k is the number of utterances. The efficient implementation is $M_L^2 M_S k^2$, where M_L is the maximum number of hypothesis that any atomic utterance can be consistent with (~ 3000), and M_S is the maximum number of atomic utterance that can be consistent with any hypothesis (49). • “referred to by "left-most"”. It should be “upper-left”. • “It wasn’t clear to me ... argmax?”. Yes it would definitely increase. Here we want to measure a more “soft” score, as Figure 5 already uses random-tie-break for white and argmax for blue. • “ user can remove a demonstration”. We observe that subjects are mostly undoing accidents. • “Did all of the human subject trials result in success?”. Subjects who did not complete the experiment were not used in the analysis.

R2, R3 Thanks for the encouragement, the two general responses should cover your concerns. We’ll use the space here to give a simple illustrative example why a prior is not enough in modeling some pragmatic behaviours. Imagine there is a race of 4 people which I attended, and I tell you “Darn I didn’t get first place”. The most likely outcome is that I placed 2nd, more likely than if I’m 3rd. However, if I tell you “Hey at least I’m not last”. The most likely outcome is that I placed 3rd, more likely than if I’m 2nd. No prior orderings of 1st,2nd,3rd,4th can model this.

R4 • “from equations 3 ... equivalent?”. The speaker model is additive: you start with 0 examples, then add 1, then add another. Equation 4 is the result of applying both equation 2 and 3, in the case that the user used examples u2, then u4, to describe hypothesis h5. • “field like [1]”. It was a great read! Their work amounts to learning a prior: rather than a prior over the programs directly, it learns a prior over the specs of the programs (i-o pairs) and implicitly ranked the programs that way. It also requires training using existing ground-truth data, which our work avoids. We will cite them.