Unified Pragmatic Models for Generating and Following Instructions

Daniel Fried Jacob Andreas Dan Klein
Computer Science Division
University of California, Berkeley
{dfried,jda,klein}@cs.berkeley.edu

Abstract

We show that explicit pragmatic inference aids in correctly generating and following natural language instructions for complex, sequential tasks. Our pragmatics-enabled models reason about why speakers produce certain instructions, and about how listeners will react upon hearing them. Like previous pragmatic models, we use learned base listener and speaker models to build a *pragmatic speaker* that uses the base listener to simulate the interpretation of candidate descriptions, and a *pragmatic listener* that reasons counterfactually about alternative descriptions. We extend these models to tasks with sequential structure. Evaluation of language generation and interpretation shows that pragmatic inference improves state-of-the-art listener models (at correctly interpreting human instructions) and speaker models (at producing instructions correctly interpreted by humans) in diverse settings.

1 Introduction

How should speakers and listeners reason about each other when they communicate? A core insight of computational pragmatics is that speaker and listener agents operate within a cooperative game-theoretic context, and that each agent benefits from reasoning about others' intents and actions within that context. Pragmatic inference has been studied by a long line of work in linguistics, natural language processing, and cognitive science. In this paper, we present a technique for layering explicit pragmatic inference on top of models for complex, sequential instruction-following and instruction-generation tasks. We investigate a range of current data sets for both tasks, showing that pragmatic behavior arises naturally from this inference procedure, and gives rise to state-of-the-art results in a variety of domains.

Consider the example shown in Figure 1a, in which a speaker agent must describe a route to a target position in a hallway. A conventional learned instruction-generating model produces a truthful description of the route (*walk forward four times*). But the pragmatic speaker in this paper, which is capable of reasoning about the listener, chooses to also include additional information (*the intersection with the bare concrete hall*), to reduce potential ambiguity and increase the odds that the listener reaches the correct destination.

This same reasoning procedure also allows a listener agent to overcome ambiguity in instructions by reasoning counterfactually about the speaker (Figure 1b). Given the command *walk along the blue carpet and you pass two objects*, a conven-

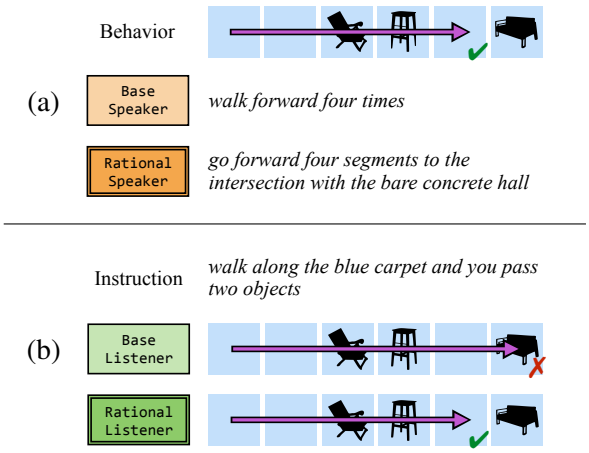


Figure 1: Real samples for the SAIL navigation environments, comparing *base* models, without explicit pragmatic inference, to the *rational* pragmatic inference procedure. (a) The rational speaker, which reasons about listener behavior, generates instructions which in this case are more robust to uncertainty about the listener's initial orientation. (b) The base listener moves to an unintended position (even though it correctly *passes two objects*). The rational listener, which reasons about the speaker, infers that a route ending at the sofa would have been described differently, and stops earlier.

tional learned instruction-following model is willing to consider all paths that pass two objects, and ultimately arrives at an unintended final position. But a pragmatic listener that reasons about the speaker can infer that the long path would have been more easily described as *go to the sofa*, and thus that the shorter path is probably intended. In these two examples, which are produced by the system we describe in this paper, a unified reasoning process (choose the output sequence which is most preferred by an embedded model of the other agent) produces pragmatic behavior for both speakers and listeners.

The application of models with explicit pragmatic reasoning abilities has so far been largely restricted to simple reference games, in which the listener's only task is to select the right item from among a small set of candidate referents given a single short utterance from the speaker. But as the example shows, there are real-world instruction following and generation tasks with rich action spaces that might also benefit from pragmatic modeling. Moreover, approaches that learn to map directly between human-annotated instructions and action sequences are ultimately limited by the effectiveness of the humans themselves. The promise of pragmatic modeling is that we can use these same annotations to build a model with a different (and perhaps even better) mechanism for interpreting and generating instructions.

The primary contribution of this work is to show how existing models of pragmatic reasoning can be extended to support instruction following and generation for challenging, multi-step, interactive tasks. Our experimental evaluation focuses on four instruction-following domains which have been studied using both semantic parsers and attentional neural models. We investigate the interrelated tasks of instruction following and instruction generation, and show that incorporating an explicit model of pragmatics helps in both cases. Reasoning about the human listener allows a speaker model to produce instructions that are easier for humans to interpret correctly in all domains (with absolute gains in accuracy ranging from 12% to 46%). Similarly, reasoning about the human speaker improves the accuracy of the listener models in interpreting instructions in most domains (with gains in accuracy of up to 10%). In all cases, the resulting systems are competitive with, and in many cases exceed, results from past state-of-the-art systems for these tasks.[1]

2 Problem Formulation

Consider the instruction following and instruction generation tasks shown in Figure 1, where an agent must produce or interpret instructions about a structured world context (e.g. *walk along the blue carpet and you pass two objects*).

In the **instruction following** task, a listener agent begins in a world state (in Figure 1 an initial map location and orientation). The agent is then tasked with following a sequence of direction sentences $d_1 \dots d_K$ produced by humans. At each time t the agent receives a percept y_t, which is a feature-based representation of the current world state, and chooses an action a_t (e.g. move forward, or turn). The agent succeeds if it is able to reach the correct final state described by the directions.

In the **instruction generation** task, the agent receives a sequence of actions $a_1, \cdots a_T$ along with the world state $y_1, \cdots y_T$ at each action, and must generate a sequence of direction sentences $d_1, \dots d_K$ describing the actions. The agent succeeds if a human listener is able to correctly follow those directions to the intended final state.

We evaluate models for both tasks in four domains. The first domain is the SAIL corpus of virtual environments and navigational directions (MacMahon et al., 2006; Chen and Mooney, 2011), where an agent navigates through a two-dimensional grid of hallways with patterned walls and floors and a discrete set of objects (Figure 1 shows a portion of one of these hallways).

In the three SCONE domains (Long et al., 2016), the world contains a number of objects with various properties, such as colored beakers which an agent can combine, drain, and mix. Instructions describe how these objects should be manipulated. These domains were designed to elicit instructions with a variety of context-dependent language phenomena, including ellipsis and coreference (Long et al., 2016) which we might expect a model of pragmatics to help resolve (Potts, 2011).

3 Related Work

The approach in this paper builds upon long lines of work in pragmatic modeling, instruction following, and instruction generation.

[1] Source code is available at http://github.com/dpfried/pragmatic-instructions

Pragmatics Our approach to pragmatics (Grice, 1975) belongs to a general category of rational speech acts models (Frank and Goodman, 2012), in which the interaction between speakers and listeners is modeled as a probabilistic process with Bayesian actors (Goodman and Stuhlmüller, 2013). Alternative formulations (e.g. with best-response rather than probabilistic dynamics) are also possible (Golland et al., 2010). Inference in these models is challenging even when the space of listener actions is extremely simple (Smith et al., 2013), and one of our goals in the present work is to show how this inference problem can be solved even in much richer action spaces than previously considered in computational pragmatics. This family of pragmatic models captures a number of important linguistic phenomena, especially those involving conversational implicature (Monroe and Potts, 2015); we note that many other topics studied under the broad heading of "pragmatics," including presupposition and indexicality, require different machinery.

Williams et al. (2015) use pragmatic reasoning with weighted inference rules to resolve ambiguity and generate clarification requests in a human-robot dialog task. Other recent work on pragmatic models focuses on the referring expression generation or "contrastive captioning" task introduced by Kazemzadeh et al. (2014). In this family are approaches that model the listener at training time (Mao et al., 2016), at evaluation time (Andreas and Klein, 2016; Monroe et al., 2017; Vedantam et al., 2017; Su et al., 2017) or both (Yu et al., 2017b; Luo and Shakhnarovich, 2017).

Other conditional sequence rescoring models that are structurally similar but motivated by concerns other than pragmatics include Li et al. (2016) and Yu et al. (2017a). Lewis et al. (2017) perform a similar inference procedure for a competitive negotiation task. The language learning model of Wang et al. (2016) also features a structured output space and uses pragmatics to improve online predictions for a semantic parsing model. Our approach in this paper performs both generation and interpretation, and investigates both structured and unstructured output representations.

Instruction following Work on instruction following tasks includes models that parse commands into structured representations processed by a rich execution model (Tellex et al., 2011; Chen, 2012; Artzi and Zettlemoyer, 2013; Guu et al., 2017), and models that map directly from instructions to a policy over primitive actions (Branavan et al., 2009), possibly mediated by an intermediate alignment or attention variable (Andreas and Klein, 2015; Mei et al., 2016). We use a model similar to Mei et al. (2016) as our base listener in this paper, evaluating on the SAIL navigation task (MacMahon et al., 2006) as they did, as well as the SCONE context-dependent execution domains (Long et al., 2016).

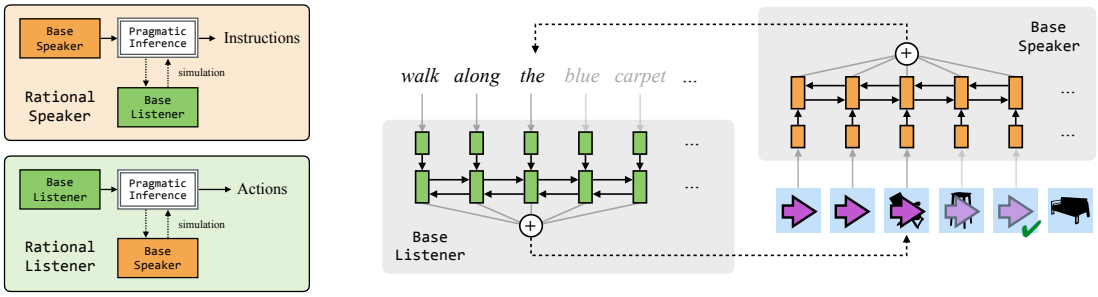
Instruction generation Previous work has also investigated the instruction generation task, in particular for navigational directions. The GIVE shared tasks (Byron et al., 2009; Koller et al., 2010; Striegnitz et al., 2011) have produced a large number of interactive direction-giving systems, both rule-based and learned. The work most immediately related to the generation task in this paper is that of Daniele et al. (2017), which also focuses on the SAIL dataset but requires substantial additional structured annotation for training, while both our base and pragmatic speaker models learn directly from strings and action sequences.

Older work has studied the properties of effective human strategies for generating navigational directions (Anderson et al., 1991). Instructions of this kind can be used to extract templates for generation (Look, 2008; Dale et al., 2005), while here we focus on the more challenging problem of learning to generate new instructions from scratch. Like our pragmatic speaker model, Goeddel and Olson (2012) also reason about listener behavior when generating navigational instructions, but rely on rule-based models for interpretation.

4 Pragmatic inference procedure

As a foundation for pragmatic inference, we assume that we have *base* listener and speaker models to map directions to actions and vice-versa. (Our notation for referring to models is adapted from Bergen et al. (2016).) The base listener, L_0, produces a probability distribution over sequences of actions, conditioned on a representation of the directions and environment as seen before each action: $P_{L_0}(a_{1:T}|d_{1:K}, y_{1:T})$. Similarly, the base speaker, S_0, defines a distribution over possible descriptions conditioned on a representation of the actions and environment: $P_{S_0}(d_{1:K}|a_{1:T}, y_{1:T})$.

Our pragmatic inference procedure requires these base models to produce candidate outputs from a given input (actions from descriptions, for



(a) (b)

Figure 2: (a) Rational pragmatic models embed base listeners and speakers. Potential candidate sequences are drawn from one base model, and then the other scores each candidate to simulate whether it produces the desired pragmatic behavior. (b) The base listener and speaker are neural sequence-to-sequence models which are largely symmetric to each other. Each produces a representation of its input sequence (a description, for the listener; actions with associated environmental percepts, for the listener) using an LSTM encoder. The output sequence is generated by an LSTM decoder attending to the input.

the listener; descriptions from actions, for the speaker), and calculate the probability of a fixed output given an input, but is otherwise agnostic to the form of the models.

We use standard sequence-to-sequence models with attention for both the base listener and speaker (described in Section 5). Our models use segmented action sequences, with one segment (sub-sequence of actions) aligned with each description sentence d_j, for all $j \in \{1 \dots K\}$. This segmentation is either given as part of the training and testing data (in the instruction following task for the SAIL domain, and in both tasks for the SCONE domain, where each sentence corresponds to a single action), or is predicted by a separate segmentation model (in the generation task for the SAIL domain), see Section 5.

4.1 Models

Using these base models as self-contained modules, we derive a *rational speaker* and *rational listener* that perform inference using embedded instances of these base models (Figure 2a). When describing an action sequence, a rational speaker S_1 chooses a description that has a high chance of causing the listener modeled by L_0 to follow the given actions:

$$S_1(a_{1:T}) = \operatorname*{argmax}_{d_{1:K}} P_{L_0}(a_{1:T}|d_{1:K}, y_{1:T}) \quad (1)$$

(noting that, in all settings we explore here, the percepts $y_{1:T}$ are completely determined by the actions $a_{1:T}$). Conversely, a rational listener L_1 follows a description by choosing an action sequence which has high probability of having caused the speaker, modeled by S_0, to produce the description:

$$L_1(d_{1:K}) = \operatorname*{argmax}_{a_{1:T}} P_{S_0}(d_{1:K}|a_{1:T}, y_{1:T}) \quad (2)$$

These optimization problems are intractable to solve for general base listener and speaker agents, including the sequence-to-sequence models we use, as they involve choosing an input (from a combinatorially large space of possible sequences) to maximize the probability of a fixed output sequence. We instead follow a simple approximate inference procedure, detailed in Section 4.2.

We consider also incorporating the scores of the base model used to produce the candidates. For the case of the speaker, we define a *combined rational speaker*, denoted $S_0 \cdot S_1$, that selects the candidate that maximizes a weighted product of probabilities under both the base listener and the base speaker:

$$\operatorname*{argmax}_{d_{1:K}} P_{L_0}(a_{1:T}|d_{1:K}, y_{1:T})^{\lambda} \times P_{S_0}(d_{1:K}|a_{1:T}, y_{1:T})^{1-\lambda} \quad (3)$$

for a fixed interpolation hyperparameter $\lambda \in [0, 1]$. There are several motivations for this combination with the base speaker score. First, as argued by Monroe et al. (2017), we would expect varying degrees of base and reasoned interpretation in human speech acts. Second, we want the descriptions produced by the model to be fluent descriptions of the actions. Since the base models are trained discriminatively, maximizing the probability of an output sequence for a fixed input sequence, their scoring behaviors for fixed outputs paired with inputs dissimilar to those seen in the training set may be

poorly calibrated (for example when conditioning on ungrammatical descriptions). Incorporating the scores of the base model used to produce the candidates aims to prevent this behavior.

To define rational listeners, we use the symmetric formulation: first, draw candidate action sequences from L_0. For L_1, choose the actions that achieve the highest probability under S_0; and for the combination model $L_0 \cdot L_1$ choose the actions with the highest weighted combination of S_0 and L_0 (paralleling equation 3).

4.2 Inference

As in past work (Smith et al., 2013; Andreas and Klein, 2016; Monroe et al., 2017), we approximate the optimization problems in equations 1, 2, and 3: use the base models to generate candidates, and rescore them to find ones that are likely to produce the desired behavior.

In the case of the rational speaker S_1, we use the base speaker S_0 to produce a set of n candidate descriptions $w^{(1)}_{1:K_1} \dots w^{(n)}_{1:K_n}$ for the sequences $a_{1:T}, y_{1:T}$, using beam search. We then find the score of each description under P_{L_0} (using it as the input sequence for the observed output actions we want the rational speaker to describe), or a weighted combination of P_{L_0} and the original candidate score P_{S_0}, and choose the description $w^{(j)}_{1:K_j}$ with the largest score, approximately solving the maximizations in equations 1 or 3, respectively. We perform a symmetric procedure for the rational listener: produce action sequence candidates from the base listener, and rescore them using the base speaker.[2]

As the rational speaker must produce long output sequences (with multiple sentences), we interleave the speaker and listener in inference, determining each output sentence sequentially. From a list of candidate direction sentences from the base speaker for the current subsequence of actions, we choose the top-scoring direction under the listener model (which may also condition on the directions which have been output previously), and then move on to the next subsequence of actions.[3]

5 Base model details

Given this framework, all that remains is to describe the base models L_0 and S_0. We implement these as sequence-to-sequence models that map directions to actions (for the listener) or actions to directions (for the speaker), additionally conditioning on the world state at each timestep.

5.1 Base listener

Our base listener model, L_0, predicts action sequences conditioned on an encoded representation of the directions and the current world state. In the SAIL domain, this is the model of Mei et al. (2016) (illustrated in green in Figure 2b for a single sentence and its associated actions), see "domain specifics" below.

Encoder Each direction sentence is encoded separately with a bidirectional LSTM (Hochreiter and Schmidhuber, 1997); the LSTM's hidden states are reset for each sentence. We obtain a representation h^e_k for the kth word in the current sentence by concatenating an embedding for the word with its forward and backward LSTM outputs.

Decoder We generate actions incrementally using an LSTM decoder with monotonic alignment between the direction sentences and subsequences of actions; at each timestep the decoder predicts the next action for the current sentence $w_{1:M}$ (including choosing to shift to the next sentence). The decoder takes as input at timestep t the current world state, y_t and a representation z_t of the current sentence, updates the decoder state h^d, and outputs a distribution over possible actions:

$$h^d_t = \text{LSTM}_d(h^d_{t-1}, [W_y y_t, z_t])$$
$$q_t = W_o(W_y y_t + W_h h^d_t + W_z z_t)$$
$$p(a_t \mid a_{1:t-1}, y_{1:t}, w_{1:M}) \propto \exp(q_t)$$

where all weight matrices W are learned parameters. The sentence representation z_t is produced using an attention mechanism (Bahdanau et al., 2015) over the representation vectors $h^e_1 \dots h^e_M$

[2] We use ensembles of models for the base listener and speaker (subsection 5.3), and to obtain candidates that are high-scoring under the combination of models in the ensemble, we perform standard beam search using all models in lock-step. At every timestep of the beam search, each possible extension of an output sequence is scored using the product of the extension's conditional probabilities across all models in the ensemble.

[3] We also experimented with sampling from the base models to produce these candidate lists, as was done in previous work (Andreas and Klein, 2016; Monroe et al., 2017). In early experiments, however, we found better performance with beam search in the rational models for all tasks.

for words in the current sentence:

$$\alpha_{t,k} \propto \exp(v \cdot \tanh(W_d h^d_{t-1} + W_e h^e_k))$$
$$z_t = \sum_{k=1}^{M} \alpha_{t,k} h^e_k$$

where the attention weights $\alpha_{t,k}$ are normalized to sum to one across positions k in the input, and weight matrices W and vector v are learned.

Domain specifics For SAIL, we use the alignments between sentences and route segments annotated by Chen and Mooney (2011), which were also used in previous work (Artzi and Zettlemoyer, 2013; Artzi et al., 2014; Mei et al., 2016). Following Mei et al. (2016), we reset the decoder's hidden state for each sentence.

In the SCONE domains, which have a larger space of possible outputs than SAIL, we extend the decoder by: (i) decomposing each action into an action type and arguments for it, (ii) using separate attention mechanisms for types and arguments and (iii) using state-dependent action embeddings. See Appendix A in the supplemental material for details. The SCONE domains are constructed so that each sentence corresponds to a single (non-decomposed) action; this provides our segmentation of the action sequence.

5.2 Base speaker

While previous work (Daniele et al., 2017) has relied on more structured approaches, we construct our base speaker model S_0 using largely the same sequence-to-sequence machinery as above. S_0 (illustrated in orange in Figure 2b) encodes a sequence of actions and world states, and then uses a decoder to output a description.

Encoder We encode the sequence of vector embeddings for the actions a_t and world states y_t using a bidirectional LSTM. Similar to the base listener's encoder, we then obtain a representation h^e_t for timestep t by concatenating a_t and y_t with the LSTM outputs at that position.

Decoder As in the listener, we use an LSTM decoder with monotonic alignment between direction sentences and subsequences of actions, and attention over the subsequences of actions. The decoder takes as input at position k an embedding for the previously generated word w_{k-1} and a representation z_k of the current subsequence of actions and world states, and produces a distribution over words (including ending the description for the current subsequence and advancing to the next). The decoder's output distribution is produced by:

$$h^d_k = \text{LSTM}_d(h^d_{k-1}, [w_{k-1}, z_k])$$
$$q_k = W_h h^d_k + W_z z_k$$
$$p(w_k \mid w_{1:k-1}, a_{1:T}, y_{1:T}) \propto \exp(q_k)$$

where all weight matrices W are learned parameters.[4] As in the base listener, the input representation z_k is produced by attending to the vectors $h^e_1 \ldots h^e_T$ encoding the input sequence (here, encoding the subsequence of actions and world states to be described):

$$\alpha_{k,t} \propto \exp(v \cdot \tanh(W_d h^d_{k-1} + W_e h^e_t))$$
$$z_k = \sum_{t=1}^{T} \alpha_{k,t}\ h^e_t$$

The decoder's LSTM state is reset at the beginning of each sentence.

Domain specifics In SAIL, for comparison to the generation system of Daniele et al. (2017) which did not use segmented routes, we train a route segmenter for use at test time. We also represent routes using a collapsed representation of action sequences. In the SCONE domains, we (i) use the same context-dependent action embeddings used in the listener, and (ii) don't require an attention mechanism, since only a single action is used to produce a given sentence within the sequence of direction sentences. See Appendix A for more details.

5.3 Training

The base listener and speaker models are trained independently to maximize the conditional likelihoods of the actions–directions pairs in the training sets. See Appendix A for details on the optimization, LSTM variant, and hyperparameters.

We use ensembles for the base listener L_0 and base speaker S_0, where each ensemble consists of 10 models trained from separate random parameter initializations. This follows the experimental setup of Mei et al. (2016) for the SAIL base listener.

[4] All parameters are distinct from those used in the base listener; the listener and speaker are trained separately.

listener	Single-sentence Rel	Single-sentence Abs	Multi-sentence Rel	Multi-sentence Abs
past work	69.98 (MBW)	65.28 (AZ)	26.07 (MBW)	35.44 (ADP)
L_0	68.40	59.62	24.79	13.53
$L_0 \cdot L_1$	**71.64**	64.38	**34.05**	24.50
accuracy gain	+3.24	+4.76	+9.26	+10.97

Table 1: Instruction-following results on the SAIL dataset. The table shows cross-validation test accuracy for the base listener (L_0) and pragmatic listeners ($L_0 \cdot L_1$), along with the gain given by pragmatics. We report results for the single- and multi-sentence conditions, under the relative and absolute starting conditions[5], comparing to the best-performing prior work by Artzi and Zettlemoyer (2013) (AZ), Artzi et al. (2014) (ADP), and Mei et al. (2016) (MBW). Bold numbers show new state-of-the-art results.

listener	Alchemy	Scene	Tangrams
GPLL	52.9	46.2	37.3
L_0	69.7	70.9	69.6
$L_0 \cdot L_1$	72.0	72.7	69.6
accuracy gain	+2.3	+1.8	+0.0

Table 2: Instruction-following results in the SCONE domains. The table shows accuracy on the test set. For reference, we also show prior results from Guu et al. (2017) (GPLL), although our models use more supervision at training time.

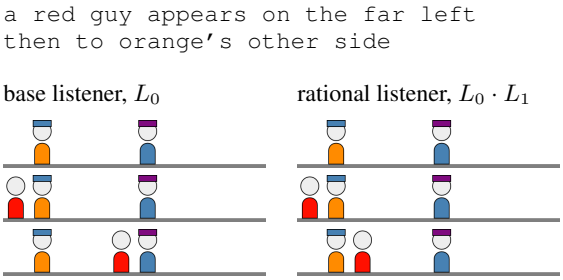


Figure 3: Action traces produced for a partial instruction sequence (two instructions out of five) in the Scene domain. The base listener moves the red figure to a position that is a marginal, but valid, interpretation of the directions. The rational listener correctly produces the action sequence the directions were intended to describe.

6 Experiments

We evaluate speaker and listener agents on both the instruction following and instruction generation tasks in the SAIL domain and three SCONE domains (Section 2). For all domains, we compare the rational listener and speaker against the base listener and speaker, as well as against past state-of-the-art results for each task and domain. Finally, we examine pragmatic inference from a model combination perspective, comparing the pragmatic reranking procedure to ensembles of a larger number of base speakers or listeners.

For all experiments, we use beam search both to generate candidate lists for the rational systems (section 4.2) and to generate the base model's output. We fix the beam size n to be the same in both the base and rational systems, using $n = 20$ for the speakers and $n = 40$ for the listeners. We tune the weight λ in the combined rational agents ($L_0 \cdot L_1$ or $S_0 \cdot S_1$) to maximize accuracy (for listener models) or BLEU (for speaker models) on each domain's development data.

6.1 Instruction following

We evaluate our listener models by their accuracy in carrying out human instructions: whether the systems were able to reach the final world state which the human was tasked with guiding them to.

SAIL We follow standard cross-validation evaluation for the instruction following task on the SAIL dataset (Artzi and Zettlemoyer, 2013; Artzi et al., 2014; Mei et al., 2016).[5] Table 1 shows improvements over the base listener L_0 when using the rational listener $L_0 \cdot L_1$ in the single- and multi-sentence settings. We also report the best accuracies from past work. We see that the largest relative gains come in the multi-sentence setting, where handling ambiguity is potentially more important to avoid compounding errors. The rational model improves on the published results of Mei et al. (2016), and while it is still below the systems of Artzi and Zettlemoyer (2013) and Artzi et al. (2014), which use additional supervision in the form of hand-annotated seed lexicons and logical domain representations, it approaches their results in the single-sentence setting.

SCONE In the SCONE domains, past work has trained listener models with weak supervision

[5]Past work has differed in the handling of undetermined orientations in the routes, which occur in the first state for multi-sentence routes and the first segment of their corresponding single-sentence routes. For comparison to both types of past work, we train and evaluate listeners in two settings: *Abs*, which sets these undetermined starting orientations to be a fixed absolute orientation, and *Rel*, where an undetermined starting orientation is set to be a 90 degree rotation from the next state in the true route.

speaker	SAIL	Alchemy	Scene	Tangrams
DBW	70.9	—	—	—
S_0	62.8	29.3	31.3	60.0
$S_0 \cdot S_1$	**75.2**	**75.3**	**69.3**	**88.0**
accuracy gain	+12.4	+46.0	+38.0	+28.0
human-generated	73.2	83.3	78.0	66.0

Table 3: Instruction generation results. We report the accuracies of human evaluators at following the outputs of the speaker systems (as well as other humans) on 50-instance samples from the SAIL dataset and SCONE domains. DBW is the system of Daniele et al. (2017). Bold numbers are new state-of-the-art results.

speaker	SAIL	Alchemy	Scene	Tangrams
DBW	11.00	—	—	—
S_0	12.04	19.34	18.09	21.75
$S_0 \cdot S_1$	10.78	18.70	27.15	23.03
BLEU gain	-1.26	-0.64	+9.06	+1.28
accuracy gain (from Table 3)	+12.4	+46.0	+38.0	+28.0

Table 4: Gains in how easy the directions are to follow are not always associated with a gain in BLEU. This table shows corpus-level 4-gram BLEU comparing outputs of the speaker systems to human-produced directions on the SAIL dataset and SCONE domains, compared to gains in accuracy when asking humans to carry out a sample of the systems' directions (see Table 3).

(with no intermediate actions between start and end world states) on a subset of the full SCONE training data. We use the full training set, and to use a model and training procedure consistent with the SAIL setting, train listener and speaker models using the intermediate actions as supervision as well.[6] The evaluation method and test data are the same as in past work on SCONE: models are provided with an initial world state and a sequence of 5 instructions to carry out, and are evaluated on their accuracy in reaching the intended final world state.

Results are reported in Table 2. We see gains from the rational system $L_0 \cdot L_1$ in both the Alchemy and Scene domains. The pragmatic inference procedure allows correcting errors or overly-literal interpretations from the base listener. An example is shown in Figure 3. The base listener (left) interprets *then to orange's other side* incorrectly, while the rational listener discounts this interpretation (it could, for example, be better described by *to the left of blue*) and produces the action the descriptions were meant to describe (right). To the extent that human annotators already account for pragmatic effects when generating instructions, examples like these suggest that our model's explicit reasoning is able to capture interpretation behavior that the base sequence-to-sequence listener model is unable to model.

6.2 Instruction generation

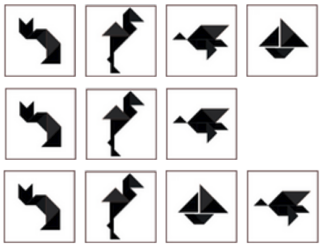
As our primary evaluation for the instruction generation task, we had Mechanical Turk workers carry out directions produced by the speaker models (and by other humans) in a simulated version of each domain. For SAIL, we use the simulator released by Daniele et al. (2017) which was used in their human evaluation results, and we construct simulators for the three SCONE domains. In all settings, we take a sample of 50 action sequences from the domain's test set (using the same sample as Daniele et al. (2017) for SAIL), and have three separate Turk workers attempt to follow the systems' directions for the action sequence.

Table 3 gives the average accuracy of subjects in reaching the intended final world state across all sampled test instances, for each domain. The "human-generated" row reports subjects' accuracy at following the datasets' reference directions. The directions produced by the base speaker S_0 are often much harder to follow than those produced by humans (e.g. 29.3% of S_0's directions are correctly interpretable for Alchemy, vs. 83.3% of human directions). However, we see substantial gains from the rational speaker $S_0 \cdot S_1$ over S_0 in all cases (with absolute gains in accuracy ranging from 12.4% to 46.0%), and the average accuracy of humans at following the rational speaker's directions is substantially higher than for human-produced directions in the Tangrams domain. In the SAIL evaluation, we also include the directions produced by the system of Daniele et al. (2017) (DBW), and find that the rational speaker's directions are followable to comparable accuracy.

We also compare the directions produced by the systems to the reference instructions given by humans in the dataset, using 4-gram BLEU[7] (Pap-

[6] Since the pragmatic inference procedure we use is agnostic to the models' training method, it could also be applied to the models of Guu et al. (2017); however we find that pragmatic inference can improve even upon our stronger base listener models.

[7] See Appendix A for details on evaluating BLEU in the SAIL setting, where there may be a different number of reference and predicted sentences for a given example.



human	take away the last item undo the last step
S_0	remove the last figure add it back
$S_0 \cdot S_1$	remove the last figure add it back in the 3rd position

Figure 4: Descriptions produced for a partial action sequence in the Tangrams domain. Neither the human nor base speaker S_0 correctly specifies where to add the shape in the second step, while the rational speaker $S_0 \cdot S_1$ does.

ineni et al., 2002) in Table 4. Consistent with past work (Krahmer and Theune, 2010), we find that BLEU score is a poor indicator of whether the directions can be correctly followed.

Qualitatively, the rational inference procedure is most successful in fixing ambiguities in the base speaker model's descriptions. Figure 4 gives a typical example of this for the last few timesteps from a Tangrams instance. The base speaker correctly describes that the shape should be added back, but does not specify where to add it, which could lead a listener to add it in the same position it was deleted. The human speaker also makes this mistake in their description. This speaks to the difficulty of describing complex actions pragmatically even for humans in the Tangrams domain. The ability of the pragmatic speaker to produce directions that are easier to follow than humans' in this domain (Table 3) shows that the pragmatic model can generate something different (and in some cases better) than the training data.

6.3 Pragmatics as model combination

Finally, our rational models can be viewed as pragmatically-motivated model combinations, producing candidates using base listener or speaker models and reranking using a combination of scores from both. We want to verify that a rational listener using n ensembled base listeners and n base speakers outperforms a simple ensemble of $2n$ base listeners (and similarly for the rational speaker).

Fixing the total number of models to 20 in each listener experiment, we find that the rational listener (using an ensemble of 10 base listener models and 10 base speaker models) still substantially outperforms the ensembled base listener (using 20 base listener models): accuracy gains are 68.5 $\rightarrow$ 71.6%, 70.1 $\rightarrow$ 72.0%, 71.9 $\rightarrow$ 72.7%, and 69.1 $\rightarrow$ 69.6% for SAIL single-sentence Rel, Alchemy, Scene, and Tangrams, respectively.

For the speaker experiments, fixing the total number of models to 10 (since inference in the speaker models is more expensive than in the follower models), we find similar gains as well: the rational speaker improves human accuracy at following the generated instructions from 61.9 $\rightarrow$ 73.4%, 30.7 $\rightarrow$ 74.7%, 32.0 $\rightarrow$ 66.0%, 58.7 $\rightarrow$ 92.7%, for SAIL, Alchemy, Scene, and Tangrams, respectively.[8]

7 Conclusion

We have demonstrated that a simple procedure for pragmatic inference, with a unified treatment for speakers and listeners, obtains improvements for instruction following as well as instruction generation in multiple settings. The inference procedure is capable of reasoning about sequential, interdependent actions in non-trivial world contexts. We find that pragmatics improves upon the performance of the base models for both tasks, in most cases substantially. While this is perhaps unsurprising for the generation task, which has been discussed from a pragmatic perspective in a variety of recent work in NLP, it is encouraging that pragmatic reasoning can also improve performance for a grounded listening task with sequential, structured output spaces.

Acknowledgments

We are grateful to Andrea Daniele for sharing the SAIL simulator and their system's outputs, to Hongyuan Mei for help with the dataset, and to Tom Griffiths and Chris Potts for helpful comments and discussion. This work was supported by DARPA through the Explainable Artificial Intelligence (XAI) program. DF is supported by a Huawei / Berkeley AI fellowship. JA is supported by a Facebook graduate fellowship.

[8] The accuracies for the base speakers are slightly different than in Table 3, despite being produced by the same systems, since we reran experiments to control as much as possible for time variation in the pool of Mechanical Turk workers.

References

Anne H. Anderson, Miles Bader, Ellen Gurman Bard, Elizabeth Boyle, Gwyneth Doherty, Simon Garrod, Stephen Isard, Jacqueline Kowtko, Jan McAllister, Jim Miller, et al. 1991. The HCRC map task corpus. *Language and speech* 34(4):351–366.

Jacob Andreas and Dan Klein. 2015. Alignment-based compositional semantics for instruction following. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Yoav Artzi, Dipanjan Das, and Slav Petrov. 2014. Learning compact lexicons for CCG semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics* 1(1):49–62.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *International Conference on Learning Representations* .

Leon Bergen, Roger Levy, and Noah Goodman. 2016. Pragmatic reasoning through semantic inference. *Semantics and Pragmatics* 9.

S.R.K. Branavan, Harr Chen, Luke S. Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 82–90.

Donna Byron, Alexander Koller, Kristina Striegnitz, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. 2009. Report on the first NLG challenge on generating instructions in virtual environments (GIVE). In *Proceedings of the 12th european workshop on natural language generation*. Association for Computational Linguistics, pages 165–173.

David L Chen. 2012. Fast online lexicon learning for grounded language acquisition. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. pages 430–439.

David L. Chen and Raymond J. Mooney. 2011. Learning to interpret natural language navigation instructions from observations. In *Proceedings of the Meeting of the Association for the Advancement of Artificial Intelligence*. volume 2, pages 1–2.

Robert Dale, Sabine Geldof, and Jean-Philippe Prost. 2005. Using natural language generation in automatic route. *Journal of Research and practice in Information Technology* 37(1):89.

Andrea F. Daniele, Mohit Bansal, and Matthew R. Walter. 2017. Navigational instruction generation as inverse reinforcement learning with neural machine translation. *Proceedings of Human-Robot Interaction* .

Michael C Frank and Noah D Goodman. 2012. Predicting pragmatic reasoning in language games. *Science* 336(6084):998–998.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems 29 (NIPS)*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*. volume 9, pages 249–256.

Robert Goeddel and Edwin Olson. 2012. Dart: A particle-based method for generating easy-to-follow directions. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, pages 1213–1219.

Dave Golland, Percy Liang, and Dan Klein. 2010. A game-theoretic approach to generating spatial descriptions. In *Proceedings of the 2010 conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 410–419.

Noah D Goodman and Andreas Stuhlmüller. 2013. Knowledge and implicature: Modeling language understanding as social cognition. *Topics in cognitive science* 5(1):173–184.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2016. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems* .

H. P. Grice. 1975. Logic and conversation. In P. Cole and J. L. Morgan, editors, *Syntax and Semantics: Vol. 3: Speech Acts*, Academic Press, San Diego, CA, pages 41–58.

Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. 2017. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Association for Computational Linguistics (ACL)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara L Berg. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. pages 787–798.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations* .

Alexander Koller, Kristina Striegnitz, Andrew Gargett, Donna Byron, Justine Cassell, Robert Dale, Johanna Moore, and Jon Oberlander. 2010. Report on the second NLG challenge on generating instructions in virtual environments (GIVE-2). In *Proceedings of the 6th international natural language generation conference*. Association for Computational Linguistics, pages 243–250.

Emiel Krahmer and Mariët Theune, editors. 2010. *Empirical Methods in Natural Language Generation: Data-oriented Methods and Empirical Evaluation*. Springer-Verlag, Berlin, Heidelberg.

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning for negotiation dialogues. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics*.

Reginald Long, Panupong Pasupat, and Percy Liang. 2016. Simpler context-dependent logical forms via model projections. In *Association for Computational Linguistics (ACL)*.

Gary Wai Keung Look. 2008. *Cognitively-inspired direction giving*. Ph.D. thesis, Massachusetts Institute of Technology.

Ruotian Luo and Gregory Shakhnarovich. 2017. Comprehension-guided referring expressions. In *Computer Vision and Pattern Recognition*.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: Connecting language, knowledge, and action in route instructions. *Proceedings of the Meeting of the Association for the Advancement of Artificial Intelligence* 2(6):4.

Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan Yuille, and Kevin Murphy. 2016. Generation and comprehension of unambiguous object descriptions. In *Computer Vision and Pattern Recognition*.

Hongyuan Mei, Mohit Bansal, and Matthew Walter. 2016. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of the Meeting of the Association for the Advancement of Artificial Intelligence*.

Will Monroe, Robert X.D. Hawkins, Noah D. Goodman, and Christopher Potts. 2017. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics* .

Will Monroe and Christopher Potts. 2015. Learning in the Rational Speech Acts model. In *Proceedings of 20th Amsterdam Colloquium*. ILLC, Amsterdam.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 311–318.

Christopher Potts. 2011. *Pragmatics*, Oxford University Press.

Nathaniel J Smith, Noah Goodman, and Michael Frank. 2013. Learning and using language via recursive pragmatic reasoning about other agents. In *Advances in Neural Information Processing Systems*. pages 3039–3047.

Kristina Striegnitz, Alexandre Denis, Andrew Gargett, Konstantina Garoufi, Alexander Koller, and Mariët Theune. 2011. Report on the second second challenge on generating instructions in virtual environments GIVE-2.5. In *Proceedings of the 13th European workshop on natural language generation*. Association for Computational Linguistics, pages 270–279.

Jong-Chyi Su, Chenyun Wu, Huaizu Jiang, and Subhransu Maji. 2017. Reasoning about fine-grained attribute phrases using reference games. In *International Conference on Computer Vision*.

Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R. Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *In Proceedings of the National Conference on Artificial Intelligence*.

Ramakrishna Vedantam, Samy Bengio, Kevin Murphy, Devi Parikh, and Gal Chechik. 2017. Context-aware captions from context-agnostic supervision. In *Computer Vision and Pattern Recognition (CVPR)*. volume 3.

Sida I. Wang, Percy Liang, and Christopher D. Manning. 2016. Learning language games through interaction. In *Association for Computational Linguistics (ACL)*.

Tom Williams, Gordon Briggs, Bradley Oosterveld, and Matthias Scheutz. 2015. Going beyond literal command-based instructions: Extending robotic natural language interaction capabilities. In *AAAI*. pages 1387–1393.

Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomas Kocisky. 2017a. The neural noisy channel. *International Conference on Learning Representations* .

Licheng Yu, Hao Tan, Mohit Bansal, and Tamara L. Berg. 2017b. A joint speaker-listener-reinforcer model for referring expressions. In *Computer Vision and Pattern Recognition*.

type	arguments	contextual embedding
Alchemy		
MIX	source i	contents of i
POUR	source i, target j	contents of i and j
DRAIN	amount a, source i	a, contents of i
Scene		
ENTER	color c, source i	people at $i-1$ and $i+1$
EXIT	source i	people at i, $i-1$, $i+1$
MOVE	source i, target j	people at i, $j-1$, $j+1$
SWITCH	source i, target j	people at i and j
TAKEHAT	source i, target j	people at i and j
Tangrams		
REMOVE	position i	—
SWAP	positions i and j	—
INSERT	position i, shape s	index of step when s was removed

Table 5: Action types, arguments, and elements of the world state or action history that are extracted to produce contextual action embeddings.

A Supplemental Material

A.1 SCONE listener details

We factor action production in each of the three SCONE domains, separately predicting the action type and the arguments specific to that action type. Action types and arguments are listed in the first two columns of Table 5. For example, Alchemy's actions involve predicting the action type, a potential source beaker index i and target beaker index j, and potential amount to drain a. All factors of the action (the type and options for each argument) are predicted using separate attention mechanisms, which produce a vector q_f giving unnormalized scores for factor f (e.g. scoring each possible type, or each possible choice for the argument).

We also obtain state-specific embeddings of actions, to make it easier for the model to learn relevant features from the state embeddings (e.g. rather than needing to learn to select the region of the state vector corresponding to the 5th beaker in the action MIX(5) in Alchemy, this action's contextual embedding encodes the current content of the 5th beaker). We incorporate these state-specific embeddings into computation of the action probabilities using a bilinear bonus score:

$$b(a) = q^\top W_{qa} a + w_a^\top a$$

where q is the concatenation of all q_f factor scoring vectors, and W_{qa} and w_a are a learned parameter matrix and vector, respectively. This bonus score $b(a)$ for each action is added to the unnormalized score for the corresponding action a (computed by summing the entries of the q_f vectors which correspond to the factored action components), and the normalized output distribution is then produced using a softmax over all valid actions.

A.2 SAIL speaker details

Since our speaker model operates on segmented action sequences, we train a route segmenter on the training data and then predict segmentations for the test data. This provides a closer comparison to the generation system of Daniele et al. (2017) which did not use segmented routes. The route segmenter runs a bidirectional LSTM over the concatenated state and action embeddings (as in the speaker encoder), then uses a logistic output layer to classify whether the route should be split at each possible timestep. We also collapse consecutive sequences of forward movement actions into single actions (e.g. MOVE4 representing four consecutive forward movements), which we found helped prevent counting errors (such as outputting *move forward three* when the correct route moved forward four steps).

A.3 SCONE speaker details

We use a one-hot representation of the arguments (see Table 5) and contextual embedding (as described in A.1) for each action a_t as input to the SCONE speaker encoder at time t (along with the representation e_t of the world state, as in SAIL). Since SCONE uses a monotonic, one-to-one alignment between actions and direction sentences, the decoder does not use a learned attention mechanism but fixes the contextual representation z_k to be the encoded vector at the action corresponding to the sentence currently being generated.

A.4 Training details

We optimize model parameters using ADAM (Kingma and Ba, 2015) with default hyperparameters and the initialization scheme of Glorot and Bengio (2010). All LSTMs have one layer. The LSTM cell in both the listener and the follower use coupled input and forget gates, and peephole connections to the cell state (Greff et al., 2016). We also apply the LSTM variational dropout scheme of Gal and Ghahramani (2016), using the same dropout rate for inputs, outputs, and recurrent connections. See Table 6 for hyperparameters. We

model	domain	dropout rate	hidden dim	attention dim
L_0	SAIL	0.25	100	100
L_0	Alchemy	0.1	50	50
L_0	Scene	0.1	100	100
L_0	Tangrams	0.3	50	100
S_0	SAIL	0.25	100	100
S_0	Alchemy	0.3	100	–
S_0	Scene	0.3	100	–
S_0	Tangrams	0.3	50	–

Table 6: Hyperparameters for the base listener (L_0) and speaker (S_0) models. The SCONE speakers do not use an attention mechanism.

perform early stopping using the evaluation metric (accuracy for the listener and BLEU score for the speaker) on the development set.

A.5 Computing BLEU for SAIL

To compute BLEU in the SAIL experiments, as the speaker models may choose produce a different number of sentences for each route than in the true description, we obtain a single sequence of words from a multi-sentence description produced for a route by concatenating the sentences, separated by end-of-sentence tokens. We then calculate corpus-level 4-gram BLEU between all these sequences in the test set and the true multi-sentence descriptions (concatenated in the same way).