A Cognitive Evaluation of Instruction Generation Agents *tl;dr* They Need Better Theory-of-Mind Capabilities

♠Lingjun Zhao* and ♣Khanh Nguyen* and ♠◇Hal Daumé III
♠University of Maryland–College Park ♣Princeton University ◇Microsoft Research
lzhao123@umd.edu

Abstract

We mathematically characterize the cognitive capabilities that enable humans to effectively guide others through natural language. We show that neural-network-based instruction generation agents possess similar cognitive capabilities, and design an evaluation scheme for probing those capabilities. Our results indicate that these agents, while capable of effectively narrowing the search space, poorly predict the listener's interpretations of their instructions and thus often fail to select the best instructions even from a small candidate set. We augment the agents with better theory-of-mind models of the listener and obtain significant performance boost in guiding real humans. Yet, there remains a considerable gap between our best agent and human guides. We discuss the challenges in closing this gap, emphasizing the need to construct better models of human behavior when interacting with AI-based agents.

1 Introduction

Instruction generation refers to the problem of guiding humans to accomplish goals through natural language. While being able to hold fluent chit-chatting conversations with humans (Thoppilan et al., 2022; OpenAI, 2022), performances of AI-based agents in this problem are still far from perfect (Zhao et al., 2021; Kojima et al., 2021; Wang et al., 2022). To build agents that communicate pragmatically like humans, we must equip them with cognitive capabilities similar to those of humans. Accomplishing this goal requires (i) mathematically characterize the capabilities that are essential for human pragmatic communication and (ii) designing an evaluation scheme for assessing these capabilities of AI-based agents.

In this paper, we present a framework for conducting fine-grained evaluation of the communication capabilities of instruction generation agents. Our evaluation focuses on cognitive capabilities that are known to be requisite for human-like pragmatic communication. The outcome of the evaluation indicates which cognitive capabilities require further development and thus can help developers direct their effort more deliberately and effectively. Figure 1 provides an overview of our approach.

To identify the cognitive capabilities essential for pragmatic communication, we build on two lines of work from socio-cognitive science: Bayesian models of cooperative communication (Wang et al., 2020; Goodman and Frank, 2016; Shafto et al., 2014) and studies on how humans implement Bayesian reasoning (Sanborn and Chater, 2016; Sanborn et al., 2010; Vul et al., 2014; Mamassian et al., 2002). These models have been shown to be capable of predicting and explaining human behaviors in various communication games. We propose a framework named *bounded pragmatic agent* that practically characterize the human cognitive process for instruction generation. We show that our framework can also describe the operation of a broad class of AI-based agents, including neural-network-based agents. Interpreting AI-based agents and humans under the same mathematical framework enables us to quantify their differences. We derive the optimality conditions that a bounded pragmatic agent must satisfy in order to generate optimally pragmatic instructions. These conditions correspond to well-known cognitive capabilities of humans: (i) the ability to efficiently generate relevant utterances (the *search* capability) (Bloom and Fischler, 1980; Gold et al., 2000; Trosborg, 2010) and (ii) the ability to accurately simulate the listener's interpretations of their utterances in the environment (the *theory-of-mind* capability) (Premack and Woodruff, 1978; Gopnik and Astington, 1988; Tomasello, 2019; Call and Tomasello, 2011). We then design an evaluation scheme for assessing these capabilities of an agent, measuring how close it is to satisfying our optimality conditions.

*The first two authors contribute equally.

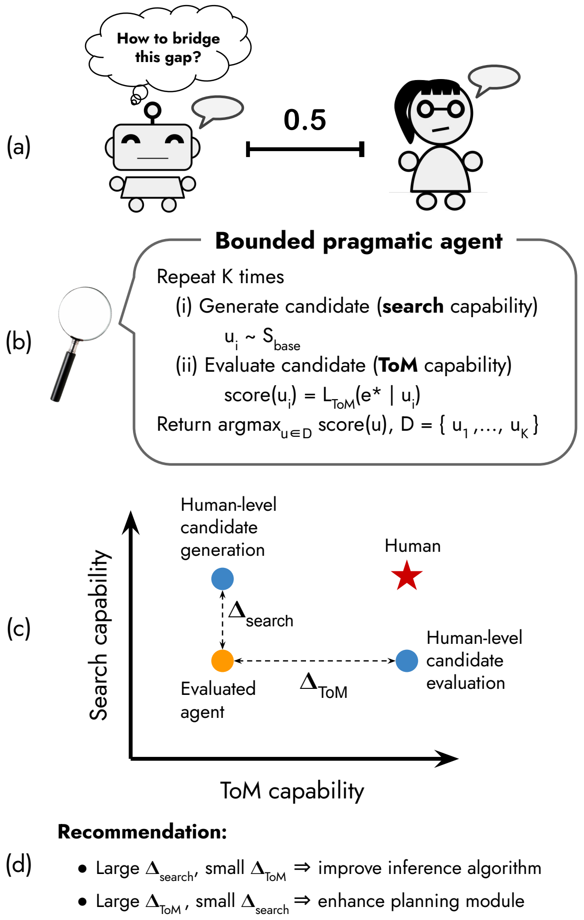

How to bridge this gap?
(a)
0.5
(b)
Bounded pragmatic agent
Repeat K times
(i) Generate candidate (**search** capability)
$u_i \sim S_{base}$
(ii) Evaluate candidate (**ToM** capability)
$score(u_i) = L_{ToM}(e^* \mid u_i)$
Return $\text{argmax}_{u \in D}$ score(u), D = { $u_1, \ldots, u_K$ }
(c)
Search capability
ToM capability
Human-level candidate generation
Human
Δ_{search}
Human-level candidate evaluation
Evaluated agent
Δ_{ToM}
(d)
Recommendation:
• Large Δ_{search}, small Δ_{ToM} ⇒ improve inference algorithm
• Large Δ_{ToM}, small Δ_{search} ⇒ enhance planning module

Figure 1: An overview of our approach. We aim to build speaker agents that can guide humans to accomplish goals through natural language. Standard evaluation that computes task performance metrics is not helpful for directing the development of the evaluated agents (a). We propose a mathematical framework called "bounded pragmatic agent" that can characterize the operations of both AI-based and human speakers (b). Viewing AI-based agents and humans through this unifying lens enables us to compare them on more fine-grained capabilities (c), and better instruct future development of these agents towards leveling with human performance (d).

We evaluate various neural-network-based agents[1] on an instruction generation problem in photo-realistic 3D environments (Anderson et al., 2018b). To evaluate each capability of an agent, we compare it with the same agent but equipped with an optimal version of the evaluated capability, which is simulated by asking a human to perform that capability for the agent. Our evaluation reveals a crucial finding: most evaluated agents possess relatively efficient search capability but inadequate theory-of-mind capability. Specifically, on a majority of test cases, the agents can find an instruction that successfully guide humans by drawing a few samples. But they assign incorrect probabilities to the instructions and thus fail to select the best one as final outputs.

We improve the theory-of-mind capability of the evaluated agents by equipping them with an explicit pragmatic reasoning mechanism (Andreas and Klein, 2016; Fried et al., 2017), using state-of-the-art instruction-following agents (Magalhaes et al., 2019; Shen et al., 2022; Hong et al., 2021) as theory-of-mind models. We obtain significant improvement over the original agents, shrinking the gap with human performance on test data by 36%. Towards eliminating the remaining gap, we illustrate with empirical evidence a major challenge in developing better theory-of-mind models. Specifically, when employed, these models would be asked to evaluate *AI-generated* instructions, which may differ dramatically from human-generated instructions. Hence, a standard supervised-learning training scheme that only exposes the model to human-generated instructions would be inadequate for learning reliable theory-of-mind models. We thus call for the construction of novel datasets, approaches, and evaluation methods for developing these models.

2 Related Work

Navigation Instruction Generation. Instruction generation has been commonly studied in navigation settings (Anderson et al., 1991; Byron et al., 2010; Koller et al., 2010; Striegnitz et al., 2011; Goeddel and Olson, 2012; Fried et al., 2017, 2018). The Matterport3D simulator and the accompanying datasets (R2R (Anderson et al., 2018b), R4R (Jain et al., 2019), and RxR (Ku et al., 2020)) offer more challenging settings by combining photo-realistic scenes with long, verbally rich instructions. Recent work on evaluating instruction generation agents (Zhao et al., 2021) reveals the ineffectiveness of standard learning and modeling approaches to this problem. Wang et al. (2021) improve the accuracy and interpretability of instructions in the RxR setting. Kamath et al. (2022) leverage this model to synthesize additional data for training instruction-following agents. Our work offers useful principles for improving these models.

[1] We release our human-evaluation dataset and interface at https://lingjunzhao.github.io/coop_instruction.html.

Mathematical Models of Human Communication. Human communication is a cooperative act (Grice, 1975; Scott-Phillips, 2014; Tomasello, 2019). Pragmatic communication in humans may involve different cognitive capabilities like basic understanding of language and social rules (Trosborg, 2010) and reasoning about the physical world (Bender and Koller, 2020) and human behavior (Ganaie and Mudasir, 2015; Enrici et al., 2019; Rubio-Fernandez, 2021). Our work describes similar capabilities but provides a mathematical interpretation that allows for computational evaluation of those capabilities. Development of mathematical models of human communication have been greatly useful for understanding human behaviors (Ho et al., 2016; Sumers et al., 2022) and building communication agents (Andreas and Klein, 2016; Fried et al., 2017, 2018; , FAIR; Lin et al., 2022). Numerous variants of these models have been proposed. Wang et al. (2020) present a comprehensive comparison of these variants and unify them under a framework inspired by optimal transport. Since we are interested more in characterizing general capabilities than specific implementation, the model we propose in this work is a generalized version capturing the essence of these models.

Evaluating Cognitive Capabilities of Neural Networks. A plethora of benchmarks for evaluating the cognitive capabilities of AI-based agents have been created, focusing on theory-of-mind capabilities (Le et al., 2019; Nematzadeh et al., 2018), grounding (Lachmy et al., 2021; Udagawa and Aizawa, 2019; Haber et al., 2019), commonsense reasoning (Talmor et al., 2018; Levesque et al., 2012; Zellers et al., 2019; Sap et al., 2019), etc. Recent work (Sap et al., 2022; Hu et al., 2022) examine performance of large language models on various cognitive tasks. They evaluate a capability by designing language tasks that are assumed to require the evaluated capability to solve. This approach is limited to large language models that can perform few-shot learning. A limitation of the approach is that it may not be possible to determine whether an agent solve the tasks in the intended way. Our evaluation scheme follows a different principle: we mathematically characterize exactly the capabilities we want to evaluate, and compare agents that possess different levels of these capabilities.

3 Problem Setting

3.1 Environment and Human Listener

We consider a human listener h acting in a POMDP environment with state space $\mathcal{S}$, action space $\mathcal{A}^h$, transition function $E^h(s_{t+1} \mid s_t, a_t)$, start-state distribution $E_1^h(s_1)$, observation space Ω, and observation function $O^h(o_{t+1} \mid s_{t+1})$. An *instruction* $\boldsymbol{u} \in \mathcal{U}$ is an utterance consisting of words belonging to a vocabulary $\mathcal{V}$. The human can follow instructions to generate trajectories. For example, in an indoor navigation setting, upon hearing "*go the kitchen and stop next to the oven*", a human can walk to the specified location. A T-step *trajectory* $\boldsymbol{e}^{\mathrm{h}} = (s_1, o_1^h, a_1^h, \cdots, s_T, o_T^h, a_T^h)$ is an execution of an instruction. The observable part of the trajectory $\bar{\boldsymbol{e}}^{\mathrm{h}} = (o_1^h, a_1^h, \cdots, o_T^h, a_T^h)$ is obtained by excluding the states from $\boldsymbol{e}^{\mathrm{h}}$.

To follow instructions, we imagine the human implements a policy $\pi^{\mathrm{h}}(a \mid \bar{\boldsymbol{e}}, \boldsymbol{u})$ that takes as input a partial observed trajectory $\bar{\boldsymbol{e}}$ and an instruction $\boldsymbol{u}$, and outputs a distribution over actions in $\mathcal{A}^h$. Given an instruction $\boldsymbol{u}$, a T-step trajectory is generated as follows. The human starts in $s_1 \sim E_1$ and observes $o_1^h \sim O(s_1)$. At time step t, let $\bar{\boldsymbol{e}}_{1:t} = (o_1^h, a_1^h, \cdots, o_t^h)$. The human chooses $a_t^h \sim \pi^{\mathrm{h}}(\cdot \mid \bar{\boldsymbol{e}}_{1:t}, \boldsymbol{u})$, executes the action, and transitions to $s_{t+1} \sim E^h(s_t, a_t^h)$. There, they perceive $o_{t+1}^h \sim O^h(s_{t+1})$. In the end, they issue a special stop action a_T to terminate the trajectory. We define $L_h(\boldsymbol{e} \mid \boldsymbol{u})$ as the probability of generating a trajectory $\boldsymbol{e}$ according to this process. We will refer to L_h as the real listener to distinguish it with the theory-of-mind listener, which is a mental model of the real listener that an agent constructs.

3.2 Pragmatic Instruction Generation

In pragmatic instruction generation (PIGEN), the goal is to learn a speaker agent r that generates language instructions to guide a human listener h to reach states in the environment. The term "pragmatic" emphasizes that the agent generates language in a social context to achieve a communication goal. In each PIGEN task, the speaker agent first imagines an intended trajectory $\boldsymbol{e}^{\star} = (s_1, o_1^r, a_1^r, \cdots, s_T, o_T^r, a_T^r)$, which leads to the intended goal state s_T from the state s_1 that the human is currently in. Because the human's action space and observation function may differ from those of the agent, they may not be able to comprehend $\boldsymbol{e}^{\star}$ even if it is presented to them. Thus, the agent needs to translate the trajectory into an *instruction* $\hat{\mathbf{u}}$ that the human can understand and follow. To do so, it implements a *speaker model* $S_r(\boldsymbol{u} \mid \boldsymbol{e})$ that takes as input a trajectory and computes a distribution over instructions. The objective of the problem can be written formally as

$$\arg\max_{S_r} \mathbb{E}_{\boldsymbol{e}^\star} \left[L_h(\boldsymbol{e}^\star \mid \mathrm{Gen}(S_r, \boldsymbol{e}^\star)) \right] \quad (1)$$

where $\mathrm{Gen}(S_r, \boldsymbol{e}^\star)$ is the process implemented by the agent for generating an instruction.

The agent is evaluated using a dataset $\mathcal{D}_{\text{eval}}$ of held-out trajectories. For each trajectory $\boldsymbol{e}^\star_k \in \mathcal{D}_{\text{eval}}$, We generate an instruction $\hat{\boldsymbol{u}}_k = \textsc{Gen}(S_r, \boldsymbol{e}^\star_k)$. The instruction is then presented to a human listener to follow, producing a trajectory $\boldsymbol{e}^{\text{h}}_k \sim L_h(\cdot \mid \hat{\boldsymbol{u}}_k)$. The performance of the agent, denoted by $\rho(r)$, is the average similarity between the human-generated trajectory and the intended trajectory

$$\rho(r) \triangleq \frac{1}{|\mathcal{D}_{\text{eval}}|} \sum_{\boldsymbol{e}^\star_k \in \mathcal{D}_{\text{eval}}} \Psi(\boldsymbol{e}^{\text{h}}_k, \boldsymbol{e}^\star_k) \quad (2)$$

where Ψ is a similarity metric.

4 Building Agents that Communicate Pragmatically like Humans

Faced with instances of the PIGEN problem daily, humans have evolved a highly efficient cognitive process for solving this problem. To build agents with a similar level of efficacy, we propose a mathematical model characterizing the human cognitive process for instruction generation (§ 4.1). We then derive the capabilities for an agent implementing that model to optimally solve PIGEN (§4.2). Finally, we present an evaluation scheme for collating these capabilities on a general class of speaker agents (§4.3).

4.1 A Mathematical Cognitive Model of Instruction Generation

To formulate how humans generate instructions, we build on mathematical models of cooperative communication (Wang et al., 2020; Goodman and Frank, 2016; Shafto et al., 2014). We consider a general version where a speaker agent constructs a *pragmatic speaker* model $S_{\text{prag}}(\boldsymbol{u} \mid \boldsymbol{e})$ based on two constituents: a *base speaker* model $S_{\text{base}}(\boldsymbol{u} \mid \boldsymbol{e})$ and a *theory-of-mind (ToM) listener* model $L_{\text{ToM}}(\boldsymbol{e} \mid \boldsymbol{u})$. The base speaker represents general knowledge of the agent about the world and the language it speaks. The ToM listener reflects situated knowledge about the listener, simulating how they would behave in the environment given an instruction. The construction of S_{prag} is defined as a Bayesian belief update that alters the initial belief S_{base} by re-weighting with L_{ToM}:

$$S_{\text{prag}}(\boldsymbol{u} \mid \boldsymbol{e}) \propto L_{\text{ToM}}(\boldsymbol{e} \mid \boldsymbol{u}) S_{\text{base}}(\boldsymbol{u} \mid \boldsymbol{e}) \quad (3)$$

The pragmatic speaker utters an instruction of maximum probability under its model:

$$\begin{aligned} \hat{\boldsymbol{u}}_{\text{prag}} &\triangleq \arg\max_{\boldsymbol{u} \in \mathcal{U}} S_{\text{prag}}(\boldsymbol{u} \mid \boldsymbol{e}^\star) \\ &= \arg\max_{\boldsymbol{u} \in \mathcal{U}} L_{\text{ToM}}(\boldsymbol{e}^\star \mid \boldsymbol{u}) S_{\text{base}}(\boldsymbol{u} \mid \boldsymbol{e}^\star) \quad (4) \end{aligned}$$

This choice reflects that the speaker wants to maximize the chance of the listener interpreting its instruction correctly, but it is still influenced by prior knowledge.

While this model accounts for human behaviors highly accurately on problems where $\mathcal{U}$ is a small discrete space (Frank and Goodman, 2012), in problems where $\mathcal{U}$ is unbounded like PIGEN, it is unlikely that humans, which are known to be agents with bounded rationality (Simon, 1957), are able to implement the full Bayesian update in the model's formulation. A hypothesis, which is supported by empirical evidence, is that humans approximate the update via Monte-Carlo sampling (Sanborn and Chater, 2016; Sanborn et al., 2010; Vul et al., 2014; Mamassian et al., 2002). Applying this hypothesis to our setting, we derive a more practical model of how human generate instructions, in which they perform the Bayesian update on a subspace $\mathcal{U}_{\text{sub}}$ of $\mathcal{U}$ chosen by drawing samples from S_{base}

$$\hat{\boldsymbol{u}}_{\text{bounded-prag}} \triangleq \arg\max_{\boldsymbol{u} \,\in\, \mathcal{U}_{\text{sub}} \,\subset\, \mathcal{U}} L_{\text{ToM}}(\boldsymbol{e}^\star \mid \boldsymbol{u}) \quad (5)$$

where $\mathcal{U}_{\text{sub}}$ is a small set of candidate instructions generated by S_{base}. We call an agent that generates instructions according to Eq 5 a *bounded pragmatic speaker* (Figure 2). For such a speaker, instruction generation involves two cognitive tasks: the candidate generation task (performed by S_{base}) and the candidate evaluation task (performed by L_{ToM}). The former task ensures that the generation of an instruction is efficient, while the latter guarantees the generated instruction conveys the intended meaning.

4.2 Essential Cognitive Capabilities of Pragmatic Instruction Generation Agents

What cognitive capabilities enable humans to effectively solve the PIGEN problem (section §3.2)?

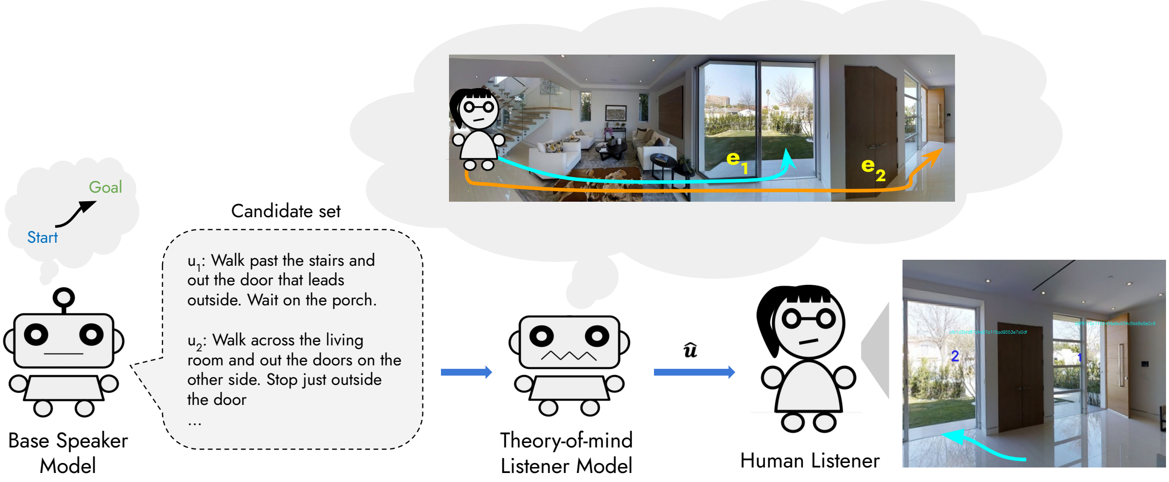

Goal
Start
Base Speaker Model
Candidate set
u$_1$: Walk past the stairs and out the door that leads outside. Wait on the porch.
u$_2$: Walk across the living room and out the doors on the other side. Stop just outside the door
...
Theory-of-mind Listener Model
$\hat{u}$
Human Listener
e$_1$
e$_2$
2

Figure 2: The cognitive process of a bounded pragmatic speaker. The speaker implements two models: a base speaker model and a theory-of-mind listener model. In every task, the speaker first imagines a trajectory it wants to convey to the human listener. To reduce the search space, it then uses the base speaker to generate a small set of relevant candidate instructions. After that, it employs the theory-of-mind model to simulate how the human listener would follow each instruction in the candidate set. The speaker finally elects the candidate instruction that causes the theory-of-mind listener to generate the trajectory most similar to the intended trajectory. The output instruction is finally sent to the human listener for a real execution in the environment.

Viewing humans as bounded pragmatic agents, we can characterize those capabilities by identifying the requirements for a bounded pragmatic agent to optimize the PIGEN objective (Eq 1). A general condition is that the instruction $\hat{\boldsymbol{u}}_{\text{bounded-prag}}$ selected by the agent must satisfy

$$\hat{\boldsymbol{u}}_{\text{bounded-prag}} = \boldsymbol{u}^\star \triangleq \arg\max_{\boldsymbol{u}} L_h(\boldsymbol{e}^\star \mid \boldsymbol{u}) \quad (6)$$

where L_h is the real listener.

We translate this condition into conditions for the constituent models, S_{base} and L_{ToM}, of the agent. The condition for S_{base} is that the candidate set $\mathcal{U}_{\text{sub}}$ generated by it must contain the optimal instruction $\boldsymbol{u}^\star$ (condition Ⓢ). Fulfilling this condition requires S_{base} to be capable of quickly generating candidates and placing sufficiently high probability on $\boldsymbol{u}^\star$ so that the instruction can be found by sampling a few candidates. We refer to this capability as the *search capability* of an agent.

The condition for L_{ToM} is that it must rank $\boldsymbol{u}^\star$ first among the candidates (condition Ⓣ). Meeting this condition demands having the capability of mentally counterfactually simulating the behavior of the listener in an environment, and evaluating whether the communicated intention is actualized in the simulation. We refer to this capability as the *ToM capability*.

The search and ToM capabilities are orthogonal and complementary. An agent with flawless ToM capability can evaluate the goodness of instructions given to it, but may not be able to efficiently generate good instructions by itself. In contrast, an agent with effective search capability can quickly bring to attention highly relevant utterances but may not always select the best one for its communication purposes if it has a misleading ToM model.

4.3 Assessing the Cognitive Capabilities of an Instruction Generation Agent

We consider a speaker agent r that learns a model $S_r(\boldsymbol{u} \mid \boldsymbol{e})$ and communicates a trajectory $\boldsymbol{e}^\star$ by running an inference algorithm to compute an instruction $\hat{\boldsymbol{u}}_{\text{infer}} \approx \arg\max_{\boldsymbol{u} \in \mathcal{U}} S_r(\boldsymbol{u} \mid \boldsymbol{e}^\star)$. Generative LSTM- or Transformer-based models that implement greedy or beam-search decoding are examples of such an agent.

We notice that, like humans, r also possesses search and ToM capabilities. On one hand, it can generate candidate instructions like a base speaker by sampling from S_r or executing an inference algorithm. On the other hand, for a fixed $\boldsymbol{e}^\star$, it can use $S_r(\boldsymbol{u} \mid \boldsymbol{e}^\star)$ as a ToM model to rank instructions. Improving these capabilities is crucial for r to better solve PIGEN. In fact, suppose S_r satisfies conditions Ⓣ and the following candidate set generated by S_r

$$\mathcal{U}^r_{\text{sub}} \triangleq \{\hat{\boldsymbol{u}}_{\text{infer}}\} \cup \{\boldsymbol{u}_i \sim S_r \mid 1 \le i \le N\} \quad (7)$$

fulfills condition Ⓢ. Then instead of running the inference algorithm, it can generate instructions as a bounded pragmatic agent as follows

$$\hat{\boldsymbol{u}} \triangleq \arg\max_{\boldsymbol{u}\in\mathcal{U}^r_{\text{sub}}} S_r(\boldsymbol{u} \mid \boldsymbol{e}^\star) \tag{8}$$

and optimizes the PIGEN objective.

To evaluate each capability of r, we measure the performance gap between the agent and a skyline agent which is at human level in the evaluated capability but is equally good as r at the other capability. Specifically, we define $r_{\text{oracle-search}}$ to be an agent that employs S_r as the ToM model but is given a "gold" candidate set $\mathcal{U}^\star_{\text{cand}}$ that always contains the ground-truth instruction $\boldsymbol{u}^\star$. It outputs an instruction as follows

$$\hat{\boldsymbol{u}}_{\text{oracle-search}} \triangleq \arg\max_{\boldsymbol{u}\in\mathcal{U}^\star_{\text{cand}}} S_r(\boldsymbol{u} \mid \boldsymbol{e}^\star) \tag{9}$$

This agent has similar ToM capability as r but human-level search capability (in fact, its search capability satisfies condition Ⓢ). Next, we construct $r_{\text{oracle-ToM}}$ which generates candidates using S_r but employs a real human to select the output instruction

$$\hat{\boldsymbol{u}}_{\text{oracle-ToM}} \triangleq \arg\max_{\boldsymbol{u}\in\mathcal{U}^r_{\text{sub}}} L_h(\boldsymbol{e}^\star \mid \boldsymbol{u}) \tag{10}$$

where $\hat{\boldsymbol{u}}_{\text{infer}}$ is the instruction generated by the inference algorithm that r implements and $\mathcal{U}^r_{\text{sub}}$ is defined as in Eq 7. The search capability of $r_{\text{oracle-ToM}}$ is as good as r but its ToM capability is that of a human.

We define the prospective performance gain (PPG) with respect to each capability as follows

$$\text{PPG}_{\text{search}}(r) \triangleq \rho(r_{\text{oracle-search}}) - \rho(r) \tag{11}$$

$$\text{PPG}_{\text{ToM}}(r) \triangleq \rho(r_{\text{oracle-ToM}}) - \rho(r) \tag{12}$$

where ρ computes the performance metric of an agent on evaluation data (Eq 2 of §3.2). The metric computes the potential improvement if one of the capability is enhanced. It implies which of the two capabilities of r is currently more deficient and thus informs future development direction for the agent. For example, if $\text{PPG}_{\text{search}}(r)$ is large and $\text{PPG}_{\text{ToM}}(r)$ is small, it means that the evaluated agent is scoring the candidate instructions highly accurately but it is bad at finding high-score instructions. In this case, developers may want to focus on devising a more effective inference algorithm for the agent. On the other hand, if the agent's estimated scores are poorly calibrated, signified by $\text{PPG}_{\text{ToM}}(r)$ being large, building a better planning module that simulates the listener's behavior more accurately would yield significant performance boost.

5 Improving ToM Capability with Ensemble Instruction-Following Agents

We improve the ToM capability of an agent r by turning it into a bounded pragmatic agent that uses the original model S_r as the base speaker but is equipped with a better ToM model than S_r. A common approach for building a ToM model is to learn an instruction-following policy $\hat{\pi}(a \mid \boldsymbol{u}, \bar{\boldsymbol{e}})$ using the same dataset used for learning S_r (Andreas and Klein, 2016; Fried et al., 2017, 2018).

We argue that this approach has a potential drawback. A ToM model learned in this way is only exposed to human-generated input instructions. At deployment time, it would likely experience a *covariate shift* because as a ToM model, the model is then asked to score instructions generated by a speaker model, not by humans. These instructions may be incorrect, ungrammatical, or may simply have a different style than human-generated instructions. This covariate shift would hamper the model's judgement. Our preliminary experiments (Appendix § A.5) confirms that using a listener trained on only human-generated inputs as the ToM model hurts rather than improves the performance of various speakers.

We show that this problem can be alleviated by employing ToM models that have calibrated uncertainty on unseen instructions. We obtain calibrated models through ensembling (Lakshminarayanan et al., 2017). Specifically, we randomly draw K 90%-samples of the training dataset. We use each sample to train an instruction-following policy $\hat{\pi}^{(k)}(a \mid \boldsymbol{u}, \bar{\boldsymbol{e}})$; the policies are also initialized with different random seeds.

When the agent has access to a simulation of the environment, it can leverage the simulation to construct better ToM models. Note that the probability that a ToM model L_{ToM} assigns to an instruction can be seen as an expectation of a 0-1 metric: $L_{\text{ToM}}(\boldsymbol{e}^\star \mid \boldsymbol{u}) = \mathbb{E}_{\boldsymbol{e}\sim L_{\text{ToM}}(\cdot\mid\boldsymbol{u})}\left[\mathbb{1}\{\boldsymbol{e} = \boldsymbol{e}^\star\}\right]$, which does not award partial credit if $\boldsymbol{e}$ partially overlaps with $\boldsymbol{e}^\star$. We make two changes: (i) replace the 0-1 metric with a soft metric $\Psi(\boldsymbol{e}, \boldsymbol{e}^\star)$ that can measure partial similarity between trajectories and (ii) approximate the expectation by executing instruction-following policies $\hat{\pi}^{(k)}$ in the environment to sample trajectories. Our final ToM-augmented agent selects its instruction as follows

$$\hat{\boldsymbol{u}}_{\text{augment-ToM}} \triangleq \arg\max_{\boldsymbol{u} \in \mathcal{U}^r_{\text{sub}}} L_{\text{ToM}}(\boldsymbol{u}, \boldsymbol{e}^\star) \tag{13}$$

$$L_{\text{ToM}}(\boldsymbol{u}, \boldsymbol{e}^\star) \triangleq \frac{1}{KM} \sum_{k=1}^{K} \sum_{j=1}^{M} \Psi(\boldsymbol{e}_j(\hat{\pi}^{(k)}, \boldsymbol{u}), \boldsymbol{e}^\star)$$

$$\mathcal{U}^r_{\text{sub}} \triangleq \{\hat{\boldsymbol{u}}_{\text{infer}}\} \cup \{\boldsymbol{u}_i \sim S_r \mid 1 \leq i \leq N\}$$

where $\boldsymbol{e}(\pi, \boldsymbol{u})$ denotes a trajectory obtained by continuously sampling actions from a policy π conditioned on an instruction $\boldsymbol{u}$.

6 Experimental Setup

6.1 Environment and Dataset

We setup an instruction generation problem in 3D environments using the Matterport3D simulator (Anderson et al., 2018b). The simulator photo-realistically emulates the visual perception of a person walking in an indoor environment. Traveling in an environment is simulated as traversing in a graph where each node corresponds to a location. At any location, an agent is provided with RGB images capturing the 360-degree panoramic view when looking from that location.

We train our speaker and listener models using the Room-to-Room (R2R) dataset which accompanies the simulator. The R2R dataset was originally created for training instruction-following agents. Each data point was collected by asking a crowd-worker to write a verbal description of a path in an environment. In the end, each path was annotated with three instructions. Each instruction contains 29 words on average. The dataset is split into a training set (61 environments, 4,675 paths), a seen validation set (340 paths) whose paths are sampled in the training environments, and an unseen validation set (11 environments unseen during training, 783 paths).

We train the models using the training set and validate them on the unseen validation set for model selection. The final performance metrics are computed on the seen validation set.

6.2 Speaker Models

We evaluate three speaker model architectures. The first is a GPT-2 model pre-trained on text (Radford et al., 2019) and fine-tuned on the R2R training set. The other two models are encoder-decoders: one implements an LSTM architecture similar to (Shen et al., 2022), and the other is based on a Transformer architecture (Vaswani et al., 2017). The parameters of these two models are randomly initialized.

Training. We train the speakers with a standard maximum-likelihood objective using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 10^{-4}. More detailed model implementation and hyperparameters are provided in § A.1. During training, we select the best model based on the unseen-validation BLEU score (Papineni et al., 2002) of the model-generated instructions with the respect to the ground-truth instructions.

6.3 Human Evaluation

We evaluate each speaker model on 75 paths in the unseen validation data split. In the end, we have annotated 1,200 instructions generated by 16 different systems (humans, 3 speaker models, and their ablated and augmented versions).

To evaluate a speaker model, we present its generated instructions to a human annotator and ask them to follow the instructions to navigate in Matterport3D environments. We adapt the PanGEA tool[2] to setup a web navigation interface and create a task on Amazon Mechanical Turk (MTurk) to recruit human evaluators. We pay the evaluator $5.20 per task which takes about 25 minutes. For each evaluation task, we ask the human evaluator to follow six instruction-following sessions.

Quality Assurance. One of the six sessions, which appears in all tasks, is a quality-control test featuring an easy-to-follow human-written instruction. We only approve an evaluator if they navigate successfully to the goal destination in this test. Following Zhao et al. (2021), we instruct the judges to not explore the environments unnecessarily and not wander back and forth unless they are lost. We record the trajectories created by the human and use them to compute the performance metrics. More details about the crowd-sourcing interface are given in Appendix §A.4.

Performance Metrics. The quality of a speaker is determined by the similarity between the intended trajectory and the actual trajectories that the

[2] https://github.com/google-research/pangea

speaker's instructions induce the human evaluators to generate. We compute these similarity metrics:

- Success rate (SR) averages binary indicators of whether the final location of a human-generated trajectory is within 3 meters of the final location of the intended trajectory;
- SPL (Anderson et al., 2018a) weights the success indicator with the ratio between the intended traveling distance and the actual one;
- NDTW and SDTW are metrics based on dynamic time-warping alignment (Magalhaes et al., 2019), capturing the similarity between two point sequences. NDTW computes only a sequence similarity score while SDTW weights the score with the success indicator.

7 Experiments

We investigate the following questions:

(a) *How well do the speakers perform on our problem?* We find that, while implementing advanced model architectures, these speakers perform poorly compared to human speakers.

(b) *What causes their performance deficiency?* Using our evaluation scheme, we identify that the speakers possess decent search capability but inadequate ToM capability.

(c) *Can we improve the speakers by equipping them with better ToM models?* We train ensembles of state-of-the-art instruction-following agents to serve as the ToM models for the speakers, and obtain significant improvements.

(d) *What are the challenges in bridging the performance gap with human speakers?* We show that state-of-the-art instruction-following agents are not optimally trained to serve as ToM models because they are mostly trained to predict how humans follow human-generated instructions, but as ToM models, they are required to accurately predict how humans follow model-generated instructions.

How well do the speakers perform on our problem? Figure 3 shows the performance of the three speaker models on variety of metrics. We also evaluate the human-written instructions provided by the R2R dataset. Overall, there is a wide margin between the models and the humans. The best model speaker (EncDec-Transformer) lags behind the humans by 21.6 NDTW points. We find that the encoder-decoder architecture with cross-attention of EncDec-Transformer outperforms the decoder-only self-attention architecture of GPT-2 (+11.7 NDTW), indicating that fusing the vision and language features too early in an architecture may be detrimental. On the other hand, EncDec-Transformer leads over EncDec-LSTM by 4.1 points, suggesting that the Transformer architecture is more effective than LSTM in this problem.

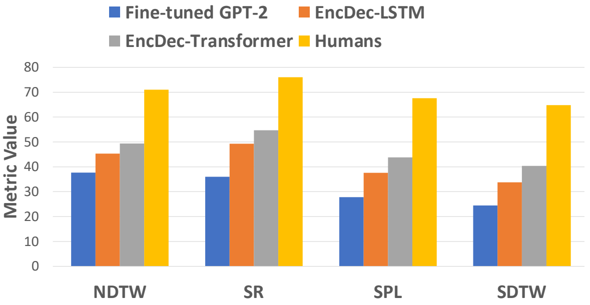


Figure 3: Performance of different speakers on held-out evaluation data. There is a considerable gap between model and humans speakers.

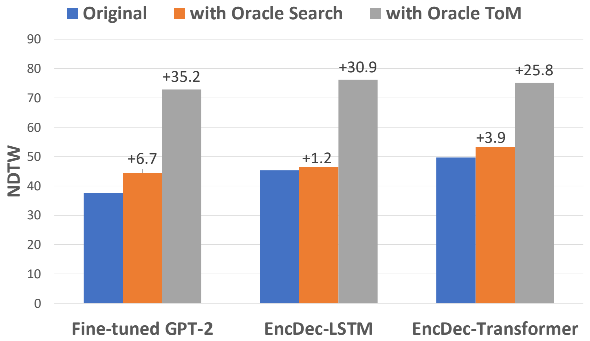


Figure 4: Performance of the speakers and their human-augmented versions. Possessing human-level ToM capability improves performance of the speakers, showing that their original ToM capability is highly deficient compared to that of humans.

What causes the speakers' performance deficiency? Next, we investigate whether the lack of search or ToM capability is responsible for the performance deficiency of the speakers. Following our evaluation scheme, we compute the prospective performance gains when one of the capabilities were made optimal. The results presented in Figure 4 show that it is an under-performed ToM capability that primarily causes the models to perform poorly. While equipping the models with optimal search capability only improves their performance by 30% on average, granting them optimal ToM capability nearly doubles their performance metrics. In fact, the search capability of the models is already as good as that of the humans we employ, because the models with optimal ToM capability achieve even

	Base speaker S_{base}		
ToM listener L_{ToM}	Fine-tuned GPT-2	EncDec-LSTM	EncDec-Transformer
None	37.7 (▲ 0.0)	45.3 (▲ 0.0)	49.4 (▲ 0.0)
Single VLN-BERT (Majumdar et al., 2020)	38.9 (▲ 1.2)	39.8 (▼ 5.5)	46.2 (▼ 3.2)
Ensemble of 10 EnvDrop-CLIP (Shen et al., 2022)	37.8 (▲ 0.1)	53.1[†] (▲ 7.8)	57.3[†] (▲ 7.9)
Ensemble of 10 VLN↻BERT (Hong et al., 2021)	43.4 (▲ 5.7)	56.4[‡] (▲ 11.1)	54.2 (▲ 4.8)
Humans (skyline)	72.9[‡] (▲ 35.2)	76.2[‡] (▲ 30.9)	75.2[‡] (▲ 25.8)

Table 1: Performance of the speakers when equipped with different ToM models. Employing ensemble instruction-following agents significantly improves their performance. [‡] and [†] indicate results that are significantly higher than those of the corresponding "None" baseline (row 1) with $p < 0.05$ and $p < 0.1$, respectively (according to a two-related-sample t-test).

	Listener		
Instructions generated by	VLN-BERT	EnvDrop-CLIP	VLN↻BERT
Humans (R2R dataset)	65.4 (▼ 0.0)	47.2 (▼ 0.0)	65.0 (▼ 0.0)
Fine-tuned GPT-2	43.1[‡] (▼ 22.3)	31.6[‡] (▼ 15.6)	39.9[‡] (▼ 25.1)
EncDec-LSTM	50.0[‡] (▼ 15.4)	43.7 (▼ 3.5)	49.3[‡] (▼ 15.7)
EncDec-Transformer	52.1[‡] (▼ 13.3)	41.5 (▼ 5.5)	51.9[‡] (▼ 13.1)

Table 2: Agreement of human and model listeners on instructions generated by different speakers. The level of agreement decreases substantially when shifting from human-generated to model-generated instructions. [‡] indicate results that are significantly lower than the human skyline (row 1) with $p < 0.05$ (according to a two-related-sample t-test).

slightly higher SDTW score than the human speakers (e.g., 75.2 of EncDec-Transformer compared to 71.0 of humans), though the differences are not statistically significant.

Can we improve the speakers by equipping them with better ToM models? Following the procedure described in Section § 5, we train various state-of-the-art instruction-following agents to serve as ToM listener models for the speakers. These listeners are trained using maximum log-likelihood on the same data as the speakers. Performances of different combinations of speakers and listeners are given in Table 1. We attain the largest improvement of 7.9 NDTW points over the best base speaker (EncDec-Transformer) by augmenting this speaker with an ensemble of 10 EnvDrop-CLIP listeners as the ToM model. We observe that ensemble models consistently outperform single models. More results about the detrimental effects of using single listeners on the speakers is given in Appendix § A.5. Despite the promising improvements, there remains a large gap of 17.9 NDTW points between our best speaker and the human speakers.

What are the challenges in bridging the performance gap with human speakers? In the previous set of experiments, a notable pattern emerges: the performance superiority of a listener on the R2R instruction-following problem, where it is asked to follow *human-generated* instruction, does not translate into a superiority in serving as a ToM model, where it is asked to rank *model-generated* instructions. To further illustrate this phenomenon, we measure the agreement between human listeners and model listeners on instructions generated by different speakers. We define the agreement score between a human L_h and a model $\hat{L}$ as

$$\text{Agreement}(L_h, \hat{L}) \tag{14}$$

$$= \text{Average}_{\boldsymbol{u} \in \mathcal{D}_{\text{eval}}} \left(\text{NDTW}(\boldsymbol{e}_h(\boldsymbol{u}), \hat{\boldsymbol{e}}(\boldsymbol{u})) \right) \tag{15}$$

where $\boldsymbol{e}_h(\boldsymbol{u})$ and $\hat{\boldsymbol{e}}(\boldsymbol{u})$ are the trajectories generated by L_h and $\hat{L}$ given $\boldsymbol{u}$, respectively, and $\mathcal{D}_{\text{eval}}$ denotes the R2R seen validation set.

As seen from Table 2, the listener agents agree more with the humans on human-generated instructions than on model-generated ones. These results can be explained through the lens of training-deployment covariate shift: during training, the model listeners are only trained to agree with human listeners on human-generated instructions and thus does not know how to behave properly on other types of instructions.

8 Conclusion

This work introduces a framework for analyzing of the cognitive capabilities of instruction generation agents. Our analysis highlights the necessity of constructing better ToM models for these agents. We argue that learning ToM models is faced with challenges that are distinct from those of learning instruction-following agents. We hope that our findings will motivate the community to create novel datasets, training methods, and evaluation procedures for tackling this problem.

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.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018a. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683.

Jacob Andreas and Dan Klein. 2016. Reasoning about pragmatics with neural listeners and speakers. *arXiv preprint arXiv:1604.00562*.

Emily M Bender and Alexander Koller. 2020. Climbing towards nlu: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 5185–5198.

Paul A Bloom and Ira Fischler. 1980. Completion norms for 329 sentence contexts. *Memory & cognition*, 8(6):631–642.

Donna Byron, Alexander Koller, Kristina Striegnitz, Justine Cassell, Robert Dale, Johanna D Moore, and Jon Oberlander. 2010. Report on the first nlg challenge on generating instructions in virtual environments (give).

Josep Call and Michael Tomasello. 2011. Does the chimpanzee have a theory of mind? 30 years later. *Human Nature and Self Design*, pages 83–96.

Ivan Enrici, Bruno G Bara, and Mauro Adenzato. 2019. Theory of mind, pragmatics and the brain: Converging evidence for the role of intention processing as a core feature ofhuman communication. *Pragmatics & Cognition*, 26(1):5–38.

Meta Fundamental AI Research Diplomacy Team (FAIR)†, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. 2022. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, page eade9097.

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

Daniel Fried, Jacob Andreas, and Dan Klein. 2017. Unified pragmatic models for generating and following instructions. *arXiv preprint arXiv:1711.04987*.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 31.

MY Ganaie and Hafiz Mudasir. 2015. A study of social intelligence & academic achievement of college students of district srinagar, j&k, india. *Journal of American Science*, 11(3):23–27.

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

Jason M Gold, Richard F Murray, Patrick J Bennett, and Allison B Sekuler. 2000. Deriving behavioural receptive fields for visually completed contours. *Current Biology*, 10(11):663–666.

Noah D Goodman and Michael C Frank. 2016. Pragmatic language interpretation as probabilistic inference. *Trends in cognitive sciences*, 20(11):818–829.

Alison Gopnik and Janet W Astington. 1988. Children's understanding of representational change and its relation to the understanding of false belief and the appearance-reality distinction. *Child development*, pages 26–37.

Herbert P Grice. 1975. Logic and conversation. In *Speech acts*, pages 41–58. Brill.

Janosch Haber, Tim Baumgärtner, Ece Takmaz, Lieke Gelderloos, Elia Bruni, and Raquel Fernández. 2019. The photobook dataset: Building common ground through visually-grounded dialogue. *arXiv preprint arXiv:1906.01530*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Mark K Ho, Michael Littman, James MacGlashan, Fiery Cushman, and Joseph L Austerweil. 2016. Showing versus doing: Teaching by demonstration. *Advances in neural information processing systems*, 29.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Jennifer Hu, Sammy Floyd, Olessia Jouravlev, Evelina Fedorenko, and Edward Gibson. 2022. A fine-grained comparison of pragmatic language understanding in humans and language models. *arXiv preprint arXiv:2212.06801*.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*.

Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. 2022. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. *arXiv preprint arXiv:2210.03112*.

Noriyuki Kojima, Alane Suhr, and Yoav Artzi. 2021. Continual learning for grounded instruction generation by observing human following behavior. *Transactions of the Association for Computational Linguistics*, 9:1303–1319.

Alexander Koller, Kristina Striegnitz, Andrew Gargett, Donna Byron, Justine Cassell, Robert Dale, Johanna D 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*. The Association for Computer Linguistics.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. *arXiv preprint arXiv:2010.07954*.

Royi Lachmy, Valentina Pyatkin, and Reut Tsarfaty. 2021. Draw me a flower: Grounding formal abstract structures stated in informal natural language. *arXiv preprint arXiv:2106.14321*.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30.

Matthew Le, Y-Lan Boureau, and Maximilian Nickel. 2019. Revisiting the evaluation of theory of mind through question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5872–5877.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Jessy Lin, Daniel Fried, Dan Klein, and Anca Dragan. 2022. Inferring rewards from language in context. *arXiv preprint arXiv:2204.02515*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Gabriel Magalhaes, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. 2019. Effective and general evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision*, pages 259–274. Springer.

Pascal Mamassian, Michael Landy, and Laurence T Maloney. 2002. Bayesian modelling of visual perception. *Probabilistic models of the brain*, 13:36.

Aida Nematzadeh, Kaylee Burns, Erin Grant, Alison Gopnik, and Tom Griffiths. 2018. Evaluating theory of mind in question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2392–2400, Brussels, Belgium. Association for Computational Linguistics.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue. `https://openai.com/blog/chatgpt`.

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 of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

David Premack and Guy Woodruff. 1978. Does the chimpanzee have a theory of mind? *Behavioral and brain sciences*, 1(4):515–526.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Paula Rubio-Fernandez. 2021. Pragmatic markers: the missing link between language and theory of mind. *Synthese*, 199(1):1125–1158.

Adam N Sanborn and Nick Chater. 2016. Bayesian brains without probabilities. *Trends in cognitive sciences*, 20(12):883–893.

Adam N Sanborn, Thomas L Griffiths, and Daniel J Navarro. 2010. Rational approximations to rational models: alternative algorithms for category learning. *Psychological review*, 117(4):1144.

Maarten Sap, Ronan LeBras, Daniel Fried, and Yejin Choi. 2022. Neural theory-of-mind? on the limits of social intelligence in large lms. *arXiv preprint arXiv:2210.13312*.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Thom Scott-Phillips. 2014. *Speaking our minds: Why human communication is different, and how language evolved to make it special*. Bloomsbury Publishing.

Patrick Shafto, Noah D Goodman, and Thomas L Griffiths. 2014. A rational account of pedagogical reasoning: Teaching by, and learning from, examples. *Cognitive psychology*, 71:55–89.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. 2022. How much can clip benefit vision-and-language tasks? In *Proceedings of the International Conference on Learning Representations*.

Herbert A Simon. 1957. Models of man; social and rational.

Kristina Striegnitz, Alexandre AJ 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 *13th European workshop on natural language generation*.

Theodore Sumers, Robert D Hawkins, Mark K Ho, Thomas L Griffiths, and Dylan Hadfield-Menell. 2022. How to talk so ai will learn: Instructions, descriptions, and autonomy. In *Advances in Neural Information Processing Systems*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Michael Tomasello. 2019. Becoming human. In *Becoming Human*. Harvard University Press.

Anna Trosborg. 2010. *Pragmatics across languages and cultures*, volume 7. De Gruyter Mouton.

Takuma Udagawa and Akiko Aizawa. 2019. A natural language corpus of common grounding under continuous and partially-observable context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7120–7127.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Edward Vul, Noah Goodman, Thomas L Griffiths, and Joshua B Tenenbaum. 2014. One and done? optimal decisions from very few samples. *Cognitive science*, 38(4):599–637.

Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. 2022. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15471–15481.

Pei Wang, Junqi Wang, Pushpi Paranamana, and Patrick Shafto. 2020. A mathematical theory of cooperative communication. *Advances in Neural Information Processing Systems*, 33:17582–17593.

Su Wang, Ceslee Montgomery, Jordi Orbay, Vighnesh Birodkar, Aleksandra Faust, Izzeddin Gur, Natasha Jaques, Austin Waters, Jason Baldridge, and Peter Anderson. 2021. Less is more: Generating grounded navigation instructions from landmarks. *arXiv preprint arXiv:2111.12872*.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6720–6731.

Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge, and Eugene Ie. 2021. On the evaluation of vision-and-language navigation instructions. *arXiv preprint arXiv:2101.10504*.

Hyperparam	GPT-2	Transformer
Learning rate	10^{-4}	10^{-4}
Batch size	4	32
Optimizer	AdamW	AdamW
Num. of training iterations	2×10^5	16×10^4
Max. action steps	15	35
Max. instruction length	100	80
Image feature size	2048	512
Orientation feature size	128	128
Embedding dropout	0.1	0.3
Hidden size	768	512
Num. of hidden layers	1	1
Hidden-layer dropout rate	0.0	0.6
Num. of encoder layers	-	2
Num. of decoder layers	12	2
Transformer dropout rate	0.1	0.3
Beam size	5	1

Table 3: Hyperparameters for training the GPT-2 EncDec-Transformer speakers.

A Appendices

A.1 Implementation of Speaker Models

The speaker models take a sequence of visual observations and actions from the trajectory $\boldsymbol{e}^\star$ as input and output a text instruction $\boldsymbol{u}$. The model is trained to estimate conditional probability $S_\theta(\boldsymbol{u}|\boldsymbol{e}^\star)$. The model and training hyperparameters are listed in Table Table 3.

Input. The input trajectory $\boldsymbol{e}^\star$ is a sequence of panoramic views and actions. Each panoramic view at time step t is represented by 36 vectors $\{o_{t,i}\}_{i=1}^{36}$, each of which is a visual feature vector extracted from a pre-trained vision model concatenated with orientation features describing the agent's current gaze direction. The image features of the GPT-2 model are extracted from a ResNet-152 model (He et al., 2016), whereas those of the encoder-decoder models are from a CLIP model (Radford et al., 2021). Each ground truth action $a_t^\star$, which moves the agent to an adjacent location, is represented by image features from the gaze direction of the agent when looking towards that adjacent location, and orientation features capturing the direction of the adjacent location relative to the agent's current gaze direction.

Output. The output of a speaker model is a language instruction describing the input trajectory. At test time, the GPT-2 model employs beam search,

	Performance Metrics					
Speaker	SR ↑	SPL ↑	NDTW ↑	SDTW ↑	Path Len ↓	Interpretability ↑
Finetuned GPT-2	36.0	27.8	37.7	24.5	20.9	2.9
EncDec-LSTM	49.3	37.6	45.3	33.8	17.4	3.3
EncDec-Transformer	54.7	43.8	49.4	40.4	15.8	3.4
Humans (R2R dataset)	76.0	67.6	71.0	64.8	14.2	3.6

Table 4: Humans evaluation results on instructions generated by the speaker models. The similarity metrics are defined in § 6.3. *Path Len* measures the average length of the generated trajectories. *Interpretability* indicates how easy or difficult to follow the instructions according to human evaluators (without knowing the ground-truth trajectory).

and the encoder-decoder models generate instructions via greedy decoding (Shen et al., 2022).

Training Objective. We train the speakers with maximum-likelihood objective:

$$\max_{\theta} \sum_{(\boldsymbol{u}^\star,\boldsymbol{e}^\star)\in\mathcal{D}_{\text{train}}} \sum_{t=1}^{|\boldsymbol{u}^\star|} \log S_\theta(\boldsymbol{u}^\star_t \mid \boldsymbol{e}^\star, \boldsymbol{u}^\star_{<t}) \tag{16}$$

where θ is the speaker model parameters, $\boldsymbol{u}^\star_t$ is t-th word of the ground-truth instruction, and $\boldsymbol{u}^\star_{<t}$ is the first $t-1$ words of the instruction.

A.2 Fine-tuning GPT-2 Speaker Model

To represent the trajectory features as a sequence of feature vectors to feed into the GPT-2 model, we first average the view features $\bar{o}_t$ for each time step:

$$\bar{o}_t = \frac{1}{36}\sum_{i=1}^{36} o_{t,i} \tag{17}$$

We compute the input features $\boldsymbol{e}^\star_t$ by concatenating the panoramic view features and ground truth action features:

$$\boldsymbol{e}^\star_t = [\bar{o}_t; a^\star_t] \tag{18}$$

The sequence of feature vectors $\boldsymbol{e}^\star$ representing a trajectory is calculated as follows

$$\boldsymbol{e}^\star = [\tanh(\boldsymbol{e}^\star_1 W); \cdots; \tanh(\boldsymbol{e}^\star_T W)] \tag{19}$$

where W is parameters of a linear layer.

For the instruction $\boldsymbol{u}^\star$, we perform an embedding look-up of its words. Then, we first prompt the model with $\boldsymbol{e}^\star$ and then train it to generate $\boldsymbol{u}^\star$ as a suffix.

A.3 Training Encoder-Decoder Models

Our EncDec-LSTM model follows the implementation of the speaker in Shen et al. (2022). We implement the EncDec-Transformer model by replacing the LSTM layers of the speaker model described in Tan et al. (2019) with Transformer layers (Vaswani et al., 2017).

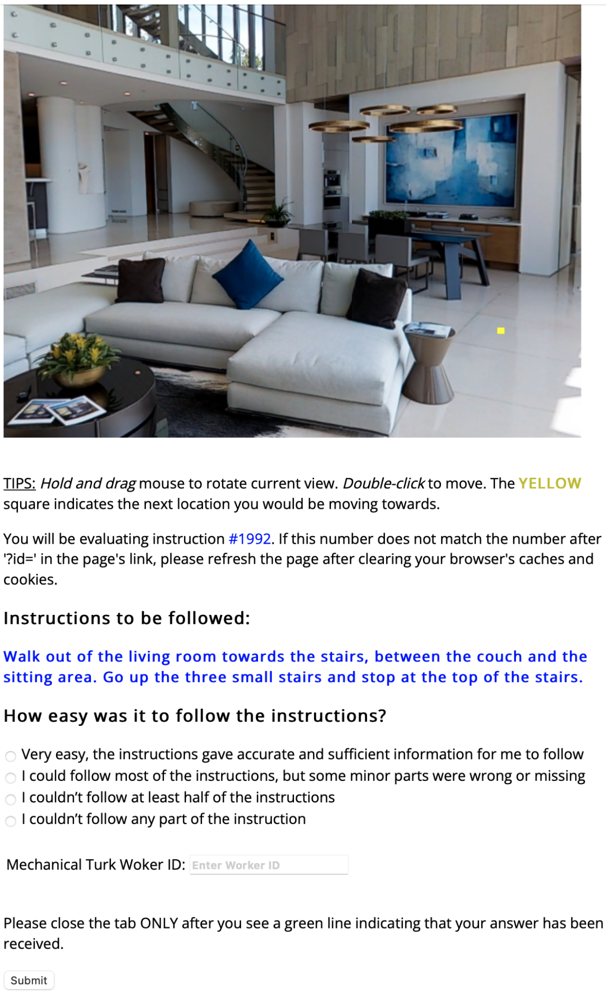
A.4 Human Evaluation Interface

Figure 5 shows the interface for our human evaluation. After a human evaluator finishes following an instruction, we recorded the path they generate and compute similarity metrics with respect to the ground-truth path. After the instruction-following sessions, we ask each evaluator to assess the interpretability of the instructions by asking them how easy (or difficult) it was for them to follow the instruction. We provide four rating levels ranging from "*1: I couldn't follow any part of the instruction*" to "*4: very easy, the instructions gave accurate and sufficient information for me to follow*". The answer of the evaluators is converted to a score between one and four.

Table 4 shows the human evaluation results of the three speaker models we evaluated.

A.5 Single vs. Ensemble Listeners

As a preliminary experiment, we compare the effectiveness of a single and an ensemble of 10 VLN↻BERT agents when serving as the ToM model of a speaker. Results in Figure 6 show that the ensemble listener is significantly better than the single listener for two different speakers.



TIPS: *Hold and drag* mouse to rotate current view. *Double-click* to move. The YELLOW square indicates the next location you would be moving towards.

You will be evaluating instruction #1992. If this number does not match the number after '?id=' in the page's link, please refresh the page after clearing your browser's caches and cookies.

Instructions to be followed:

Walk out of the living room towards the stairs, between the couch and the sitting area. Go up the three small stairs and stop at the top of the stairs.

How easy was it to follow the instructions?

○ Very easy, the instructions gave accurate and sufficient information for me to follow
○ I could follow most of the instructions, but some minor parts were wrong or missing
○ I couldn't follow at least half of the instructions
○ I couldn't follow any part of the instruction

Mechanical Turk Woker ID: Enter Worker ID

Please close the tab ONLY after you see a green line indicating that your answer has been received.

Figure 5: Human evaluation interface.

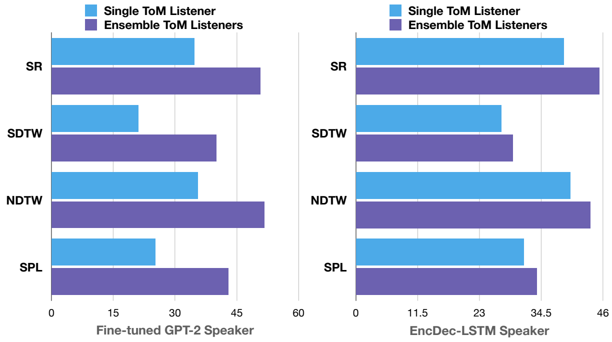


Figure 6: Comparison of single and ensemble ToM listeners.