Retrospective Learning from Interactions

Zizhao Chen, Mustafa Omer Gul, Yiwei Chen, Gloria Geng, Anne Wu & Yoav Artzi
Department of Computer Science and Cornell Tech, Cornell University
{czz,momergul,annewu,yoav}@cs.cornell.edu {yc833,gcg46}@cornell.edu

Abstract

Multi-turn interactions between large language models (LLMs) and users naturally include implicit feedback signals. If an LLM responds in an unexpected way to an instruction, the user is likely to signal it by rephrasing the request, expressing frustration, or pivoting to an alternative task. Such signals are task-independent and occupy a relatively constrained subspace of language, allowing the LLM to identify them even if it fails on the actual task. We introduce RESPECT, a method to learn from such signals in past interactions via retrospection without additional annotations. We deploy RESPECT in a new multimodal interaction scenario, where humans instruct a multimodal LLM to solve an abstract reasoning task with a combinatorial solution space. Through thousands of interactions with humans, we show how RESPECT gradually improves task completion rate from 31% to 82%, all without any external annotation.

1 Introduction

Multi-turn interactions between human users and large language models (LLMs) are naturally rich with implicit learning cues. If the LLM fails to respond appropriately, the user may express frustration, rephrase their intent, or even completely pivot what they ask for. If the LLM does well, the user may express approval or simply continue to their next objective. This is not a property unique to LLMs, but a general characteristic of effective natural language communication (Leavitt and Mueller, 1951). Such signals can inform the LLM of its performance, thereby creating an opportunity to learn through deployment, with no annotation cost.

We introduce RESPECT, a method for a model to learn from its own past interactions with human users. Rather than relying on feedback from annotators (Ouyang et al., 2022) or assuming access to stronger models (Bai et al., 2022), RESPECT relies solely on regular deployment interactions, where users interact with the model to achieve their goals. The key is using the deployment model (i.e., not a stronger or specialized model) to decode the implicit feedback expressed in follow-up human utterances in its own past interactions, thereby allowing the model to autonomously bootstrap from its interactions with human users.

We experiment with RESPECT by deploying a multimodal LLM (MLLM) in MULTIREF, a new multi-turn grounded interaction scenario (Section 3). MULTIREF is a challenging generalization of reference games (Rosenberg and Cohen, 1964) in that it requires humans to instruct the model step by step and models to reason about abstract visuals. While RESPECT is designed for broad and domain-independent deployment, MULTIREF provides an ideal test bed to answer our research questions in a lab environment. It naturally elicits the kind of gradual interactions common in conversational interactions with LLMs. It poses a challenging task to contemporary MLLMs, allowing us to observe a significant learning effect. At the same time, it is scoped, enabling a strong effect with limited data.

The key insight underlying RESPECT is that conversational implicit feedback signals occupy a relatively constrained subspace of natural language. Such signals can include direct approvals (e.g., *great!*) or signs of frustration (e.g., *not again*), and also more subtle cues, such as when the user rephrases their request. Critically, it is relatively simple to disentangle them from task performance. A human can easily figure out from such cues if they do well or not, even if they have little understanding about what they are asked for. It is this constrained nature that makes reasoning about such signals to be within the capacities of LLMs, even if they fail at the task at hand.

Figure 1 illustrates the RESPECT process: the model interacts with humans to accomplish tasks, retrospectively examines its own past interactions, and then re-trains. This process progresses in

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 24580–24606
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

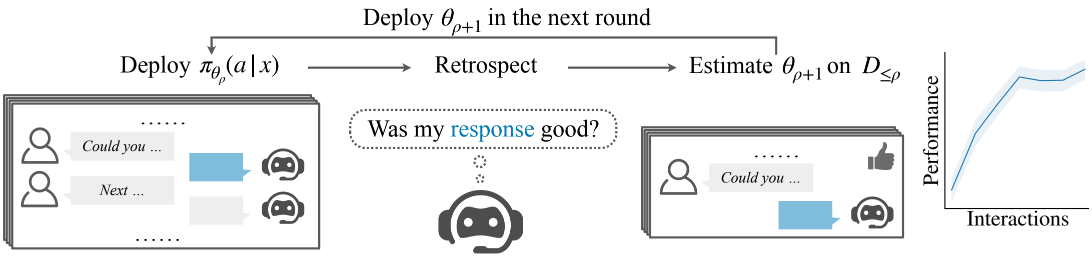


Figure 1: The RESPECT process. We deploy an MLLM policy $\pi_{\theta_\rho}(a|x)$ in rounds ρ, to interact with users in multi-turn interactions. Following each round, the MLLM retrospectively analyzes each of its actions (highlighted in blue) to decode feedback given the interaction context and follow-up utterances. The decoded feedback can be positive (thumbs up as illustrated), negative or neutral. After each round, the model is retrained using all data aggregated so far $D_{\leq\rho}$. The MLLM improves over time without any external annotations. The plot on the right shows the performance curve in our experiments – task success rate improves from 31% to 82% over six rounds.

rounds, alternating between interaction and training, improving model task capability over time. Critically, unlike common recipes for training from human feedback, RESPECT does not require any external annotation (Ouyang et al., 2022, RLHF) or even soliciting feedback from the users themselves (Suhr and Artzi, 2023).

We deploy RESPECT in MULTIREF over multiple rounds of grounded interactions with human users and re-training. We use IDEFICS2-8B (Laurençon et al., 2024) as our MLLM, and experiment with multiple learning algorithms, including filtered fine-tuning (FFT), REINFORCE-style policy gradient (Williams, 1992; Kojima et al., 2021), and KTO (Ethayarajh et al., 2024). Across our experiments, we observe that IDEFICS2-8B effectively decodes feedback, even as it initially performs poorly at the task. In our longest running experiment, we observe model task completion rate improves from 31% to 82%. Our code, data, and models are available at `https://lil-lab.github.io/respect`.

2 Technical Overview and Notation

We conduct our study in an interaction scenario called MULTIREF (Section 3). Each *interaction* is a *game* that involves multiple turns. At each *turn*, the speaker first produces a free-form natural language text, and then the listener performs an *action* according to a *policy*. Zooming out of the mechanics of the interaction, we experiment with a continual learning setup (Section 4). Our study progresses in *rounds*. In each round, the MLLM is first deployed to interact with users and complete tasks, and then the interactions are used to re-train the policy's model. Multiple rounds enable us to observe the long-term dynamics of learning from the model's own interactions. This includes the robustness of our feedback decoding and training methods to the changing distribution of the data likely to be seen in an adaptive system in the wild.

Task Notation The MLLM policy's task is to respond effectively to human utterances given in conversational context. Formally, let $\pi_\theta(a_t|x_t)$ be the θ-parameterized MLLM policy, with t being the current interaction turn, a_t an action string that represents the model response, and x_t being the context on which the policy is conditioned. The context includes the interaction history up to and excluding turn t, including current (i.e., at turn $t-1$) and past user utterances, as well as any other relevant context in which the interaction takes place.

Learning and Deployment Each round ρ includes a deployment, followed by training on the interactions between the deployed model and humans (Figure 1). During deployment at round ρ, the model π_{θ_ρ} interacts with users. For each model action $\hat{a}_t \sim \pi_{\theta_\rho}(a|x_t)$, we record a tuple $(x_t, \hat{a}_t, p_t, \bar{f}_t)$, where x_t is the context given to the model at time t to predict action $\hat{a}_t$, p_t is the probability of $\hat{a}_t$ at the time of prediction, and $\bar{f}_t$ is the remainder of the interaction following $\hat{a}_t$. Critically, we do not solicit human feedback during the interaction or after it. We compute the implicit feedback $\hat{\gamma}_t$ using a feedback decoder $\phi(x_t, \hat{a}_t, \bar{f}_t)$, to obtain tuples $(x_t, \hat{a}_t, \hat{\gamma}_t, p_t)$. We experiment with three learning objectives using this feedback: filtered fine-tuning (FFT), policy gradient, and KTO.

Evaluation We measure the quality of the model $\pi_{\theta_\rho}(a_t|x_t)$ at each round ρ primarily by interaction success rates from live human-model deployments. The same interactions are used to train the model for the next round. We track other metrics, such as the number of turns per interaction as an efficiency measure. We also annotate a subset of the inter-

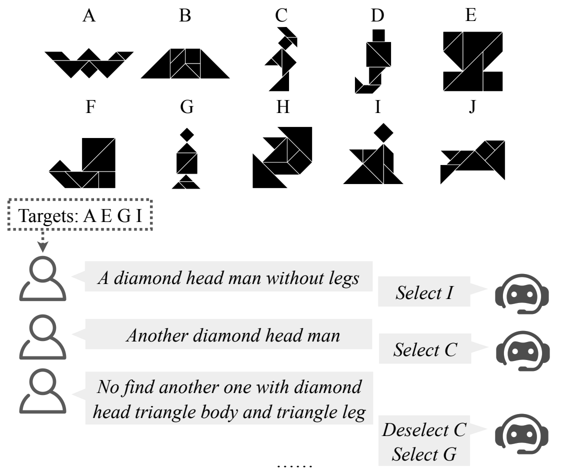


Figure 2: The interaction scenario we use in our experiments. MULTIREF is a multi-turn reference game. A speaker and a listener both observe a shared set of tangram shapes in different orders. The speaker describes targets for the listener to select, often gradually over multiple turns. As an interaction progresses, the speaker naturally produces implicit feedback signals that validate or reject the listener's actions.

actions post hoc to measure utterance-level policy success rate and feedback decoder accuracy.

3 MULTIREF: a Multi-turn Grounded Interaction Scenario

Key to our study is that tasks are relayed gradually across multiple turns, as often occurs in human interaction. We create MULTIREF, a conversational interaction scenario where two partners, a *speaker* and a *listener*, coordinate on the selection of a set of items (Figure 2). In our studies, the speaker is always a human, and the listener is a model. MULTIREF generalizes the commonly studied reference game scenario. Its design and our choice of stimuli are grounded in existing work from both cognitive science and computational language modeling (Rosenberg and Cohen, 1964; Clark and Wilkes-Gibbs, 1986; Schober and Clark, 1989; Goodman and Frank, 2016). Both partners observe a shared set of images, but in different order. The speaker is given a subset of the images as targets, with the goal of communicating the targets to the listener, so the latter selects the exact subset. Only the speaker can write natural language messages and only the listener can select or deselect images, by generating a structured string (e.g., `Deselect E select F`). The interaction concludes successfully once all and only targets are selected, or fails on timeout, 20 turns in our studies.

MULTIREF is both accessible to crowdsourcing workers and encourages constructing the solution in multiple turns, thereby creating multi-turn interactions that likely include the learning signals we aim to study. The rules are simple: the speaker describes targets for the listener to select. This makes MULTIREF easily accessible to crowdsourcing workers. At the same time, the solution the speaker communicates is relatively complex, because of the large solution space. In conventional reference games, the listener's goal is to select one image from n images, so the number of possible solutions is n. In MULTIREF, the goal is to select a subset of unknown size from n images, so the combinatorial solution space is exponential in n. Meanwhile, the solution is decomposable, creating natural opportunities for gradual instruction and implicit, immediate, and incremental feedback.

We use tangram shapes from the diverse KILOGRAM dataset (Ji et al., 2022). Tangrams are abstract shapes that are designed to elicit common concepts in humans. This abstractness often leads to ambiguous descriptions open to interpretation, e.g., Shape A in Figure 2 can be described as a *bat*, a *lowercase w*, or even a *star wars star fighter*. Tangrams naturally provide an ambiguous and challenging stimuli for human interaction (Clark and Wilkes-Gibbs, 1986; Schober and Clark, 1989; Fox Tree, 1999; Hawkins et al., 2020b), thereby leading to highly diverse language. They also remain challenging for contemporary MLLMs to reason about (Ji et al., 2022; Cheng et al., 2024), leaving significant room for learning.

The free-form natural language human speakers produce in MULTIREF is very diverse, and balances between competing pressures. First, it often requires complex pragmatic reasoning (Clark and Wilkes-Gibbs, 1986; Schober and Clark, 1989; Horton and Gerrig, 2002), because of the abstractness of tangrams. This is compounded by how the combinatorial solution space drives humans to balance between relaying as much information as possible, and relaying clear objectives to make gradual progress – a balance between two Gricean maxims: quantity and manner (Grice, 1975). Speakers may include explicit feedback such as *good*, or *deselect the last one*; the speaker may describe two targets in a single utterance (*select two men*); speakers may refer to previous selections without directly describing targets (*the other one* or *try again*). Appendix H shows several interaction case studies. Diverse language with abstract stimuli poses a challenging reasoning problem for the listener model.

MultiRef is not designed to arbitrarily increase complexity, but to naturally expose core aspects of human communication. Yet the scenario is controlled and scoped, allowing for easy measurement of task completion and progress, and making learning feasible with relatively limited crowdsourcing. This makes MultiRef particularly suitable for research in academia or other low-resource settings.

4 ReSpect: Retrospective Learning from Past Interactions

ReSpect has two components: decoding implicit feedback from past interactions (*retrospection*) and learning from the decoded feedback signals (*learning*). We deploy ReSpect in an iterative continual learning setup to observe the dynamics of learning from the models' own interaction over time. However, the method itself is not limited to continual learning, and can be applied a single step.

Formally, ReSpect re-estimates the policy parameters θ given on-policy interactions. We assume access to a raw dataset $D^{\text{raw}} = \{(x^{(i)}, \hat{a}^{(i)}, p^{(i)}, \bar{f}^{(i)})\}_{i=1}^{N}$, where $x^{(i)}$ is the policy context consisting of images and the conversation so far, $\hat{a}^{(i)}$ is the predicted action, $p^{(i)}$ is the probability of this action, and $\bar{f}^{(i)}$ is the remainder of the interaction following $\hat{a}^{(i)}$.[1] A single interaction is split into multiple training datapoints.[2] In our continual learning setup, D^{raw} is a union of all data collected from past rounds. The feedback decoder ϕ computes a categorical feedback $\hat{\gamma}^{(i)} \in \{\texttt{positive}, \texttt{neutral}, \texttt{negative}\}$ for each action $\hat{a}^{(i)}$ holistically based on its context $x^{(i)}$, action taken $\hat{a}^{(i)}$, and follow-up utterances $\bar{f}^{(i)}$.[3] This process transforms D^{raw} to $D = \{(x^{(i)}, \hat{a}^{(i)}, p^{(i)}, \hat{\gamma}^{(i)})\}_{i=1}^{N}$.

4.1 Decoding Implicit Feedback

We implement the feedback decoder ϕ by prompting the model to analyze past interaction tuples $(x, \hat{a}, p, \bar{f})$ to compute feedback $\hat{\gamma} = \phi(x, \hat{a}, \bar{f})$ (Figure 3). We hypothesize that pretrained LLMs have the ability to reason about the relatively constrained space of implicit feedback signals, even if they fail at the task. Critically, this process does not rely on a stronger LLM for critique nor on past interactions created by other LLMs, ruling out concerns about distillation. We experiment with binary or ternary feedback. The feedback decoder is designed to identify general linguistic cues, and not for the specific task we study. We assume no access to any auxiliary annotation or privileged information (e.g., no access to selection ground-truth or interaction success). This assumption of no auxiliary labels or human annotation enables us to explore the most general scenario of learning from interactions, and our method scales freely as deployment data streams in.

> User: Please carefully read the following conversation and answer: Is the very last utterance from the speaker positive or negative positive, neutral, or negative feedback? Often negative feedback include corrections and keywords like no, not, undo, don't, with generally negative sentiment, while positive feedback often includes good, yes, correct, okay, or simply move on to the next stage. Lean towards negative if it sounds neutral.
> (start of the conversation)
> Listener: Deselect F select G
> Speaker: yes, pick the thin person with a triangle head
> Listener: Select A .... *(Action to focus on)*
> Speaker: yes, pick the house with chimney .. *(Feedback)*
> (end of the conversation)
> Answer a single word, Positive, or Negative Positive, Neutral or Negative.
> Assistant: **Positive**

Figure 3: The text-only prompt used to decode feedback from past interactions. This figure combines the prompts for both binary and ternary feedback decoding. Green: binary case only. Orange: ternary case only. The verbal **feedback generated by the model** is in bold. Additional *comments for readability* are in blue italics.

4.2 Learning

We study three approaches to learn from the processed dataset $D = \{(x^{(i)}, \hat{a}^{(i)}, p^{(i)}, \hat{\gamma}^{(i)})\}_{i=1}^{N}$.

Filtered Fine-Tuning We fine-tune on positive data points ($\hat{\gamma}^{(i)} = \texttt{positive}$) and reject data points with decoded `neutral` or `negative` feedback. We use cross entropy loss, with label smoothing to regularize learning. This method follows approaches that filter to retain examples to reinforce, but relies on decoded feedback, rather than, for example, verifiers (Zelikman et al., 2024).

Reinforcement Learning We use simple REINFORCE-style policy gradient (Williams, 1992). This choice has been shown as effective in few-sample regimes, as in studies with human users (Kojima et al., 2021; Suhr and Artzi, 2023). It avoids data demanding value function estimation and plurality of hyperparameters, both downsides of algorithms like PPO (Schulman et al., 2017).

[1] For simplicity, we omit the turn index in this section.

[2] Multiple selections or deselections in one turn such as `Deselect E select F` are considered a single action.

[3] We do not compute feedback for the last action in each interaction because there is no follow-up interaction. They are excluded from D^{raw}.

Recent work (Ahmadian et al., 2024) also suggests REINFORCE can produce on-par results in LLMs with PPO despite its simplicity. Because of the few-sample regime, we do not have sufficient data to estimate a reward function, so cannot perform online RL (Ouyang et al., 2022). We train in an offline fashion within each individual round. The iterative rounds of training and deployment create a hybrid offline-online process.

We map the categorical feedback generated by the decoder $\gamma^{(i)}$ to numerical rewards:

$$R(\gamma) = \begin{cases} 1, & \gamma = \texttt{positive} \\ 0, & \gamma = \texttt{neutral} \\ -0.1, & \gamma = \texttt{negative} \end{cases} . \quad (1)$$

Dropping the i-superscripts for simplicity, the gradient estimator for a single example is:

$$\begin{aligned} \frac{\partial}{\partial \theta} &= cR(\hat{\gamma})\nabla \log P(\hat{a}|x;\theta) \\ c &= \begin{cases} 1, & \text{if } R(\hat{\gamma}) \geq 0 \\ \frac{P(\hat{a}|x;\theta)}{p}, & \text{if } R(\hat{\gamma}) < 0 \end{cases} , \end{aligned} \quad (2)$$

where the coefficient c downweights examples with negative reward by their inverse propensity score (Kojima et al., 2021). This is critical because $\lim_{P(\cdot)\to 0} \log P(\cdot) = -\infty$.

Utility Maximization We use Kahneman-Tversky Optimization (KTO; Ethayarajh et al., 2024). KTO fits our scenario because it assumes per-example binary human feedback, in contrast to methods like DPO that require pair-wise preferences (Rafailov et al., 2023). We consider examples with decoded `positive` feedback as *desired* utterances, those with decoded `negative` feedback as *undesired*, and discard those with `neutral` feedback. We refer readers to Ethayarajh et al. (2024) for the definition of the objective.

5 Experimental setup

Interaction Instantiation We use the KILOGRAM (Ji et al., 2022) tangram images (Gul and Artzi, 2024). KILOGRAM contains 1,013 images. We randomly split them into a main split (912 tangrams) and a development split (101 tangrams). We create interaction contexts by randomly sampling 10 tangrams, and randomly selecting 3–5 as targets (Appendix A). The development split is exclusively used for seeding the initial listener policy π_{θ_0}, and all human-bot interactions are conducted on images from the main split, i.e., tangrams that the seed policy π_{θ_0} has never seen before.

Model and Initialization We use IDEFICS2-8B (Laurençon et al., 2024) for both the policy and feedback decoder. We fine-tune with LoRA (Hu et al., 2022). We seed the initial policy π_{θ_0} by fine-tuning the pretrained IDEFICS2-8B weights on a small dataset from 25 human-human games constructed with the development split tangrams. Appendix B.2 provides further details. D_0 is reused in continual training via rehearsal. We use the original IDEFICS2-8B for feedback decoding, because our limited data is likely to inhibit general linguistic knowledge. This means we cannot benefit from improvements in the model feedback decoding, likely low-balling the potential of the approach.[4] The original IDEFICS2-8B provides robust feedback decoding out of the box, confirming our hypothesis, and providing a solid ground for our experiments.

System Variants We study six system variants along two dimensions: feedback decoder configuration (binary B vs. ternary T) and optimization methods (FFT vs. RL vs. KTO): B-FFT and T-FFT train on positive data points with an FFT objective (FFT). B-RL and T-RL trains on both positive and negative data points using RL. B-KTO and T-KTO are like B-RL and T-RL, but using KTO.

For variants involving negative data points (B-RL, T-RL, B-KTO, and T-KTO), we subsample negative ones to keep the positive:negative ratio close to 5:4 (Ethayarajh et al., 2024).

Deployment We conduct three rounds of training-deployment for all six systems and three more rounds for the top system, B-FFT, to observe its progress over a longer period. This cascaded design is a direct consequence of the high cost of crowdsourcing. We do not distinguish between training and evaluation in the traditional sense. Instead, all listener policies are evaluated live on MTurk on about 330 human-bot interactions each round containing roughly 2,400 turns. The same data is used to train the next iteration of policies. The policies in the same round are deployed concurrently in a randomized experiment on the same set of games to mitigate bias due to game difficulty. Appendix A.3 provides further details.

Learning Implementation Details We use the validation set for model selection throughout continual learning. Following prior work (Misra et al., 2017; Müller et al., 2019; Liu et al., 2022), we add an entropy term and length normalization to

[4] It remains an important direction for future work to keep the decoder model in sync with the policy.

all three objectives to reduce over-fitting given the relatively small amount of data. Appendix B provides additional reproducibility details. Unlike FFT and RL, where we train from scratch each round, when using KTO, we continually fine-tune from a previous model checkpoint θ_ρ to obtain $\theta_{\rho+1}$ with data accumulation. This was shown to outperform training from scratch in pilot studies.

Evaluation We evaluate each system variant at each round by the success rate during the live deployment. We report both interaction- and utterance-level success rates. The interaction level success rate is straightforward - whether the game ended with all targets selected by the listener and nothing else. We do not have access to utterance-level ground truth (i.e., the intended action) to compute success rate, so we sample 1,000 utterances per round from B-FFT to annotate by MTurk workers post-hoc. We report two measures: exact match between the annotation and model action and similarity score, which is based on the computed similarity between the tangrams selected or deselected during the turn by the human annotator and the system. We evaluate the feedback decoder by comparing its predictions with human interpretations collected during the post-hoc annotation. Lastly, we track the number of turns per interaction. Appendix B.4 provides metric definitions.

6 Results and analysis

We deploy our models for three rounds, with additional three rounds for B-FFT, the best-performing variant, to better understand long-term dynamics. All our results are from concurrent randomized deployments, where the models interact with humans in real time. We collect a total of 7,230 interactions consisting of 55,004 utterances over four weeks, at a cost of \$11,180 USD.[5]

Figure 4 shows deployment statistics for all six system variants, as well as control deployments for the initial policy and human-human games.[6] Figure 5 shows utterance-level statistics for B-FFT from the post-hoc annotations we collected for evaluation. The interaction success rate of all systems improves monotonically in the first three rounds,

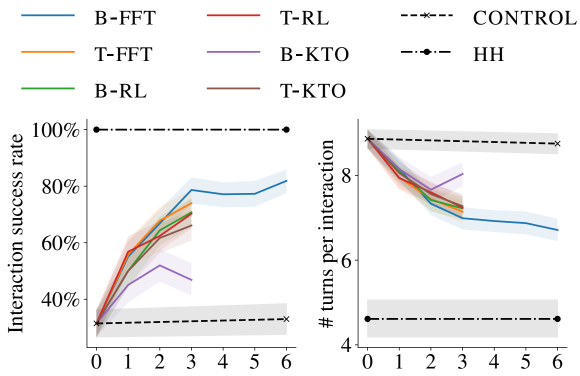


Figure 4: Task performance and efficiency improve as the policy learns from more past interactions. We present deployment results across three rounds for six concurrent systems, and three more rounds for the best system B-FFT, together with human-human references (HH) and a redeployment of the initial policy π_{θ_0} (CONTROL). *Left:* interaction-level success rate (↑, higher is better). *Right:* interaction-level efficiency by # turns per interactions (↓). Shades are 95% confidence intervals by bootstrapping with 10,000 resamples.

except for B-KTO in round 3. B-FFT plateaus in rounds 4 and 5, before resuming its improvement.[7] Figures 21–23 in the appendix show example final-round interactions comparing the initial and final models, as well as human listeners.

Overall, B-FFT improves interaction-level success rate by 51% (31%→82%) and utterance-level exact match by 22% (31%→53%). At the last round, following the plateau, B-FFT interaction success rate improves by 5% (77%→82%). The number of turns follows these trends. As the policy gets better, more games are completed within the allotted number of turns, and even faster. B-FFT starts with 8.9 turns per game, and concludes with 6.7 per game. The center panel of Figure 5 shows that actions taken by the policy increasingly resemble human actions, even mistakes (actions that receive negative feedback) become more similar to human actions. All other statistics largely track these, except some of the utterance-level statistics around when B-FFT plateaus. While all show a deviation from the monotonous earlier trend, some show a temporary decrease and not just a stagnation, but delayed by one round. This illustrates the complex

[5]This crowdsourcing cost is for conducting the controlled experiment. There are no data costs when applying RESPECT on a deployed system, because learning signals arise from interactions, not from external annotations.

[6]We present results in rounds for simplicity. Appendix C connects rounds to cumulative number of interactions. Appendix E presents full tables corresponding to these plots.

[7]The reasons behind the plateau are hard to infer. One hypothesis is that changes in the amount of data over time made some settings sub-optimal. We conducted a separate deployment, branching out from round 3 with B-FFT and more expressive LoRA adapters. This increase in expressivity allows the model to continue its monotonous improvement (Appendix D). This mini experiment illustrates the complexities of continual learning with current learning systems.

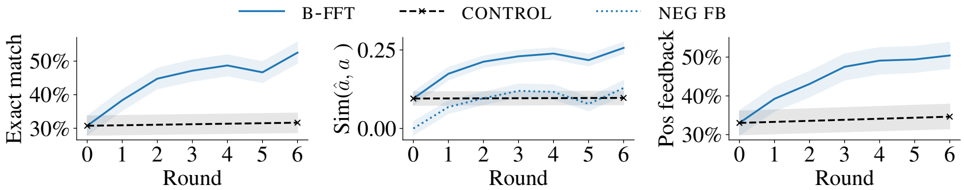


Figure 5: Turn-level performance of B-FFT evaluated by post-hoc human annotations. *Left:* % turns where the policy's action $\hat{a}$ matches exactly the human listener's action a^* (↑). *Center:* similarity between the policy's action and the human listener's action (↑). Even actions that receive negative feedback in deployment (NEG FB) are increasingly similar to human actions. *Right:* % policy actions annotated to have received positive implicit feedback from human listeners (↑). Shades are 95% confidence intervals by bootstrapping with 10,000 resamples.

dynamics of continual learning.

There remains a gap between B-FFT (our leading system) and HH (human-human interactions) even with the same worker pool. HH interactions show perfect task success rate and high efficiency. Our intuition is that the gap is due to the lack of long-term credit assignment in our learning method. This is especially influential in learning to reason about later turns, where credit assignment to the longer history is more complex. Excluding some past turns (i.e., sliding window approach) may address this issue. This learning challenge is compounded by data scarcity: we have significantly less data for later turns.

User Adaptation A potential confounding factor to the improvement in interaction success rate is users adapting to the interaction scenario and the model, instead of policy improvement (Hawkins et al., 2020a). We redeploy the initial policy π_{θ_0} concurrently in the final B-FFT round to test this (CONTROL in Figure 4). The interaction success rate of CONTROL remains unchanged over time (31% → 33%), suggesting speaker adaptation do not explain the overall 51% absolute improvement in B-FFT task success rate.

Positive Only vs. All Data The difference between systems using positive learning signals only (B-FFT, T-FFT) and those using all (B-RL, T-RL, B-KTO, T-KTO) is in learning objectives (FFT/RL/KTO). Overall, the systems based on positive signals only perform better. It is expected that positive signals will be more informative for learning. Our policy acts in a large action space. Negative rewards suppress specific actions, but without more information about what a good action is, they simply encourage a uniform distribution. This has been shown to have a helpful regularizing effect in Kojima et al. (2021). However, not only does negative feedback not help meaningfully, it seems

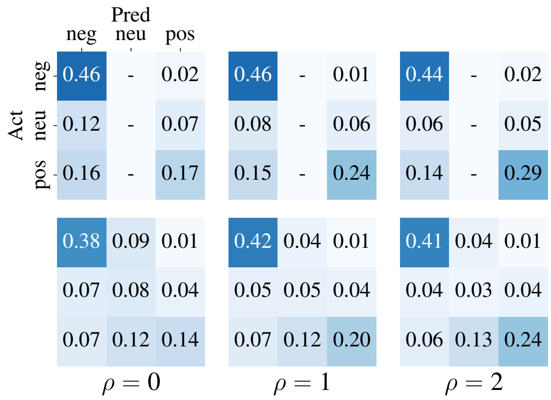


Figure 6: Confusion matrices of the binary (top) and ternary (bottom) feedback decoders over rounds. Feedback decoders yield negligibly low false positives (top right corner). The feedback decoder also correctly classifies more than 60% (diagonals) across rounds.

to confuse the learner. The positive-only systems that, in effect, have access to fewer learning signals perform better. Utilizing negative signals better is an important direction for future work.

Feedback Decoder Quality We evaluate the feedback decoder through our post-hoc annotation task. Workers annotate each turn if the speaker was satisfied with the answer given their follow-up utterances. The feedback decoder performance is stable throughout the rounds, showing robustness to changes in the data distribution (Figure 6). We observe above 90% precision consistently, after combining actual positives and neutrals. The ternary feedback decoder is more conservative and labels more positive turns as neutrals. This is a task-dependent trade-off. The zero feedback of neutrals essentially eliminates the examples, but allows for slightly cleaner data. We empirically observed it is beneficial to have slightly noisy data but more of it.

FFT vs. REINFORCE vs. KTO Overall, the FFT (B-FFT and T-FFT) perform best. The KTO variants (B-KTO and T-KTO) trail after the REIN-

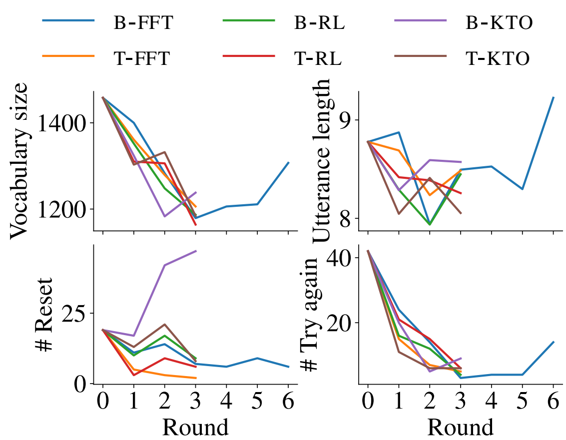


Figure 7: Language analysis of human instructions. All systems see a decrease in instruction complexity in the first three rounds, except for B-KTO, suggesting adaptation on the speaker's side, as expected when humans become more familiar with the domain. Reset/frustration signals drop, a reflection of the model improving.

FORCE variants (B-RL, T-RL). B-KTO even diverges at some point. We suspect that the KTO recipe struggles in the continual optimization scenario, where the model is fine-tuned multiple times. We observe that B-KTO deteriorates in rounds 2 and 3, and starts generating illegal outputs (e.g., `Deselect select`). Appendix B.3 describes a quick intervention we applied to mitigate this issue. Although it eliminated the illegal outputs, the quality remained low. Further refinement of KTO or hyperparameters may help, however, this is a complex process in a live deployment.

Language Analysis Human instructions changes over time (Figure 7). We observe a reduction in vocabulary size and utterance length early on. This is expected, and follows known observations in how humans adapt to reduce cognitive costs (e.g., Clark and Wilkes-Gibbs, 1986; Effenberger et al., 2021). However, in later rounds, B-FFT witnesses an increase in vocabulary size and utterance length. This surprising trend reversal is attributed to three outlier workers, so does not reflect a change in population behavior. Reset signals drop, an indication of improved collaborated task performance. These trends are fairly consistent across system variants, except for B-KTO, which also diverges in performance. Initially workers tend to use *Try again* instead of directly describing a target, or request a reset with instructions like *Deselect everything* (Figure 17 and Figure 18). The occurrences of both decrease in later rounds. Even though workers change their language, this does not influence the initial policy's π_{θ_0} performance (Figure 4).

7 Related work

Learning from Feedback RL from human feedback (RLHF; Ouyang et al., 2022), the most common recipe to learn from feedback, relies on soliciting pair-wise preferences from annotators, while we rely on *unpaired* signals from the interaction itself. Learning from explicit feedback on a single output has also been studied, either in the form of binary feedback (Ethayarajh et al., 2024; Suhr and Artzi, 2023; Gao et al., 2023) or through more expressive editing (Gao et al., 2024) or commenting and refinement (Li et al., 2017; Sumers et al., 2021; Scheurer et al., 2023). Hancock et al. (2019) pauses interactions by consulting a separately trained satisfaction predictor, and solicits explicit feedback. We do not solicit explicit feedback, but rely on natural signals that arise from the follow-up interaction.

Learning from Naturally Occurring Signals Kojima et al. (2021) learns to generate instructions by observing how humans follow them. Pang et al. (2023) maximizes heuristics, such as longer responses from humans. Artzi and Zettlemoyer (2011) studied the use of naturally occurring recovery efforts. Instead, we opt for a general approach to infer feedback from natural interactions. Concurrent work by Don-Yehiya et al. (2024) uses similar linguistic cues to ours, basing their reward decoder on the taxonomy of Petrak et al. (2023). They focus on a distillation-like scenario, where the interactions they learn from originate from other models, many of which are stronger than the model they train. We focus on self-improvement, where it is critical that no stronger model is involved. Our works complement each other and strengthen our conclusions. Their work shows our signal of interest can be derived from large-scale diverse data, whereas we show a single-model loop can use this signal to drive improvement over time.

LLMs that Self-improve A common approach to improve models is via AI feedback, solicited from the model itself or another model (Bai et al., 2022; Burns et al., 2023; Madaan et al., 2023; Kumar et al., 2024; Qu et al., 2024; Yuan et al., 2024; Li et al., 2024), or via a hand-crafted verifier (Anil et al., 2021; Kirchner et al., 2024). In contrast, we rely on implicit *real human* feedback from deployment interactions. This signal is less influenced by model biases or limitations, and does not require a model to validate the task. Our approach can be combined with any of the above, and allows to leverage deployment interactions for free.

8 Conclusion

We study the ability of models to decode implicit feedback from interactions with humans, and the efficacy of this learning signal. We operationalize learning from this signal as retrospective learning, an annotation-free approach that leverages signals from naturally occurring feedback in interactions. We demonstrate its effectiveness in long-term human-in-the-loop deployments and robustness to variants. We hope to unveil the potential of a common yet underutilized learning signal and eventually inspire an evolving language model that learns continuously without any expert annotation.

Limitations

We design MULTIREF to study real interactions over a period of time, as opposed to evaluating on a static benchmark. We make trade-offs between the generality of the task, its fit to our experimental questions, and the ability to iterate on a prototype fast, and without high costs. MULTIREF exposes the problem we aim to study and provides an experimental ground to show a strong, measurable effect, without requiring prohibitive amount of data. Any such data requirement would make our deployment impossible in research settings, thereby not allowing us to observe the dynamics of interacting with real humans over a period of time. That said, the cost of this choice is that our models specialize to the task of MULTIREF. They do not improve their abilities in ways that generalize beyond MULTIREF, and likely even experience an erosion of these capabilities. While our setup allows us to provide a clean and significant answer to our research questions, it is important to expand our study to other domains, such as summarization or conversational question answering, where similar signals may be more complex, farther apart, or demand long-term credit assignment. Our method uses only a scalar reward. Another interesting orthogonal direction is expanding the expressivity of the feedback decoder, such that it recovers a more expressive signal (e.g., a natural language explanation).

Ethical considerations

The ultimate application of our method is an automated pipeline that continuously learns from human-model interactions in deployment without additional human annotations. A naive implementation of our method, like any other approach that learns from human feedback data in deployment, is subject to data poisoning and potentially learning harmful behaviors when, for example, a malicious user verbally incentivizes harmful answers. In general, data quality assurance is essential to safeguard learning from human feedback in deployment.

Acknowledgments

This research was supported by NSF under grants No. 1750499 and OAC-2311521, NASA under award No. 20-OSTFL20-0053, a gift from Open Philanthropy, a gift from Apple, the National Artificial Intelligence Research Resource (NAIRR) Pilot, the Frontera supercomputer supported by the National Science Foundation (award NSF-OAC 1818253) at the Texas Advanced Computing Center (TACC) at The University of Texas at Austin, and the Delta advanced computing and data resource which is supported by the National Science Foundation (award NSF-OAC 2005572). GG was supported by an NSF REU supplement for NSF grant No. 175049. YC was supported by Bowers Undergraduate Research Experience program. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, NASA, or the other funders.

Author Contributions ZC developed and implemented the methods, trained models, conducted pilot studies, and the continual learning study. MOG contributed code for tangram reference games and designed the human evaluation study. MOG, AW, YC, and GG designed the MULTIREF scenario. YC and GG implemented the MULTIREF interface, recruited MTurk workers, and collected human-human interactions, with MOG and AW's mentorship. ZC, MOG, and AW contributed to the writing. YA advised and oversaw all parts of the project.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12248–12267, Bangkok, Thailand. Association for Computational Linguistics.

Abdullah Almaatouq, Joshua Becker, James P. Houghton, Nicolas Paton, Duncan J. Watts, and

Mark E. Whiting. 2021. Empirica: A virtual lab for high-throughput macro-level experiments. *Behavior Research Methods*, 53(5):2158–2171.

Cem Anil, Guodong Zhang, Yuhuai Wu, and Roger Grosse. 2021. Learning to Give Checkable Answers with Prover-Verifier Games. *Preprint*, arXiv:2108.12099.

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping semantic parsers from conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 421–432, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022. Constitutional AI: Harmlessness from AI Feedback. *Preprint*, arXiv:2212.08073.

Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeff Wu. 2023. Weak-to-Strong Generalization: Eliciting Strong Capabilities With Weak Supervision. *Preprint*, arXiv:2312.09390.

Zihui Cheng, Qiguang Chen, Jin Zhang, Hao Fei, Xiaocheng Feng, Wanxiang Che, Min Li, and Libo Qin. 2024. Comt: A novel benchmark for chain of multi-modal thought on large vision-language models. *arXiv preprint arXiv:2412.12932*.

Herbert H Clark and Deanna Wilkes-Gibbs. 1986. Referring as a collaborative process. *Cognition*, 22(1):1–39.

Shachar Don-Yehiya, Leshem Choshen, and Omri Abend. 2024. Learning from Naturally Occurring Feedback. *Preprint*, arXiv:2407.10944.

Anna Effenberger, Rhia Singh, Eva Yan, Alane Suhr, and Yoav Artzi. 2021. Analysis of language change in collaborative instruction following. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2803–2811, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. KTO: Model Alignment as Prospect Theoretic Optimization. *Preprint*, arXiv:2402.01306.

Jean E Fox Tree. 1999. Listening in on monologues and dialogues. *Discourse processes*, 27(1):35–53.

Ge Gao, Hung-Ting Chen, Yoav Artzi, and Eunsol Choi. 2023. Continually improving extractive QA via human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 406–423, Singapore. Association for Computational Linguistics.

Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. 2024. Aligning LLM Agents by Learning Latent Preference from User Edits. *Preprint*, arXiv:2404.15269.

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

Herbert Paul Grice. 1975. Logic and conversation. *Syntax and semantics*, 3:43–58.

Mustafa Omer Gul and Yoav Artzi. 2024. CoGen: Learning from Feedback with Coupled Comprehension and Generation.

Braden Hancock, Antoine Bordes, Pierre-Emmanuel Mazare, and Jason Weston. 2019. Learning from dialogue after deployment: Feed yourself, chatbot! In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3667–3684, Florence, Italy. Association for Computational Linguistics.

Robert Hawkins, Minae Kwon, Dorsa Sadigh, and Noah Goodman. 2020a. Continual Adaptation for Efficient Machine Communication. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 408–419, Online. Association for Computational Linguistics.

Robert D. Hawkins, Michael C. Frank, and Noah D. Goodman. 2020b. Characterizing the Dynamics of Learning in Repeated Reference Games. *Cognitive Science*, 44(6):e12845.

William S. Horton and Richard J. Gerrig. 2002. Speakers' experiences and audience design: Knowing *when* and knowing *how* to adjust utterances to addressees. *Journal of Memory and Language*, 47(4):589–606.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Anya Ji, Noriyuki Kojima, Noah Rush, Alane Suhr, Wai Keen Vong, Robert Hawkins, and Yoav Artzi. 2022. Abstract Visual Reasoning with Tangram Shapes. In *Proceedings of the 2022 Conference on*

Empirical Methods in Natural Language Processing, pages 582–601, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jan Hendrik Kirchner, Yining Chen, Harri Edwards, Jan Leike, Nat McAleese, and Yuri Burda. 2024. Prover-Verifier Games improve legibility of LLM outputs. *Preprint*, arXiv:2407.13692.

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.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, et al. 2024. Training language models to self-correct via reinforcement learning. *arXiv preprint arXiv:2409.12917*.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. 2024. What matters when building vision-language models? *Preprint*, arXiv:2405.02246.

Harold J Leavitt and Ronald AH Mueller. 1951. Some effects of feedback on communication. *Human relations*, 4(4):401–410.

Jiwei Li, Alexander H. Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. 2017. Dialogue learning with human-in-the-loop. In *International Conference on Learning Representations*.

Ming Li, Lichang Chen, Jiuhai Chen, Shwai He, Jiuxiang Gu, and Tianyi Zhou. 2024. Selective reflection-tuning: Student-selected data recycling for LLM instruction-tuning. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 16189–16211, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. 2022. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Dipendra Misra, John Langford, and Yoav Artzi. 2017. Mapping instructions and visual observations to actions with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1004–1015, Copenhagen, Denmark. Association for Computational Linguistics.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. 2018. Ray: A distributed framework for emerging AI applications. In *Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation*, OSDI'18, pages 561–577, USA. USENIX Association.

Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. 2019. When does label smoothing help? *Advances in neural information processing systems*, 32.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Richard Yuanzhe Pang, Stephen Roller, Kyunghyun Cho, He He, and Jason Weston. 2023. Leveraging Implicit Feedback from Deployment Data in Dialogue. *Preprint*, arXiv:2307.14117.

Dominic Petrak, Nafise Moosavi, Ye Tian, Nikolai Rozanov, and Iryna Gurevych. 2023. Learning from free-text human feedback – collect new datasets or extend existing ones? In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16259–16279, Singapore. Association for Computational Linguistics.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. 2024. Recursive introspection: Teaching LLM agents how to self-improve. In *ICML 2024 Workshop on Foundation Models in the Wild*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Seymour Rosenberg and Bertram D. Cohen. 1964. Speakers' and Listeners' Processes in a Word-Communication Task. *Science*, 145(3637):1201–1203.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. 2023. Training Language Models with Language Feedback at Scale. *Preprint*, arXiv:2303.16755.

Michael F Schober and Herbert H Clark. 1989. Understanding by addressees and overhearers. *Cognitive Psychology*, 21(2):211–232.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *Preprint*, arXiv:1707.06347.

Alane Suhr and Yoav Artzi. 2023. Continual learning for instruction following from realtime feedback. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Theodore R. Sumers, Mark K. Ho, Robert D. Hawkins, Karthik Narasimhan, and Thomas L. Griffiths. 2021. Learning Rewards From Linguistic Feedback. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7):6002–6010.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason E Weston. 2024. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*.

Eric Zelikman, Georges Harik, Yijia Shao, Varuna Jayasiri, Nick Haber, and Noah D Goodman. 2024. Quiet-star: Language models can teach themselves to think before speaking. *arXiv preprint arXiv:2403.09629*.

A The MULTIREF Game Design and Data Collection

A.1 Interaction Design

MULTIREF is a multi-target, multi-turn reference game between two players, a *speaker* and a *listener*. Each game starts with 10 tangrams as the *context*, with 3–5 tangrams designated as *targets*. The target designations are revealed to the speaker but hidden to the listener. The goal is to select all targets without selecting any non-targets. The speaker can only communicate with the listener through a sequence of utterances, and only the listener can take selection and deselection actions. The interaction starts with a speaker turn. Turns alternate between speaker and listener, with a maximum of 20 turns. In each speaker turn, they type an utterance to send to the listener. Speaker turns are limited to 25 seconds. In each listener turn, they have 45 seconds to select or deselect images as instructed to by the speaker. The game concludes when the listener selects only and all targets, or when the partners run out of turns. Appendix A.3 shows screenshots of the interface.

Context Construction We follow Gul and Artzi (2024) and construct game contexts using 1,013 tangram images from KILOGRAM (Ji et al., 2022). We group tangrams randomly into two splits: development split (101 tangrams) and main split (912 tangrams). The development split is exclusively used for seeding the initial listener policy π_{θ_0}. All human-bot interactions are constructed from the main split, i.e., tangrams that the seed policy π_{θ_0} has never seen before. We construct all games with 3–5 target tangrams. More targets are generally harder, given the same maximum number of turns per interaction.

A.2 Human Evaluation Design

Automatically evaluating turn-level policy performance is hard, because we have no ground truth (i.e., the selection and deselection actions intended by the speaker in each turn) to compare against. Similarly, we have no ground truth to systematically assess the feedback decoder quality. We conduct human evaluation surveys to address these problems. We annotate a subset of B-FFT interactions, roughly 120 interactions or 1,000 turns per system-turn.

We show human annotators a complete interaction turn by turn, without revealing the underlying targets. For each turn, the annotation consists of two phases:

1. Ground-truth: we show context, currently selected tangrams, and instruction given by the speaker. We ask the annotator to annotate the listener action. The annotator action a^* is considered as ground truth action for this turn. We use these labels for turn-level evaluation. After the action annotation, we reveal the action $\hat{a}$ actually taken by the listener (i.e., the model) during the interaction.

2. Satisfaction: we present the follow-up utterance. We ask the annotator to rate if the speaker is satisfied with the listener's action, based on the follow-up utterance. They choose one of the following options:

   a. Yes.
   b. Yes, even though the listener did not perform all required selections/deselections.
   c. Yes, even though the listener made incorrect selections/deselections.
   d. No.

   The third option accounts for the listener accidentally selecting a target tangram not intended by the speaker, but the speaker choosing to move on without correction or even validating the selection. We treat these labels as ground truth for evaluating feedback decoders.

We annotate 5% of long-term human-bot interactions annotations by three different annotators, to estimate how reliable the annotations are. We observe 85% agreement on the correctness (whether $\hat{a} = a^*$) on ground truth stage,[8] and 65% agreement on the ground-truth action a^* across workers.[9] For satisfaction annotation, we observe 93% agreement rate, illustrating the simplicity of extracting the signal that drives our learning process.

A.3 MTurk Details

Worker Recruitment We follow Gul and Artzi's (2024) worker recruitment recipe. We require workers to have a minimum 98% approval rate, at least 1,000 approved HITs (Human Intelligence Task), and be located in English-majority locales. All

[8] The percentage of cases where all annotators agree that the bot did right or wrong.

[9] The percentage of cases where all three annotators provided exactly the same set of actions.

workers must watch a video tutorial and pass a quiz before gaining qualification to work on MULTIREF interactions. They must read a thorough guideline and pass another quiz before being granted access to human evaluation surveys. We recruit 33 expert workers to interact with LLMs in the main study and annotate by completing surveys after the main study. They are required to accept a consent form detailing how MTurk worker IDs are encrypted, how the collected data would be published, and risks of participating in this study. This study is exempted from Institutional Review Board.

Payment We pay workers \$0.81 USD per MULTIREF game, and a bonus if the game is successful. Overall the estimated hourly wage is \$13.00 USD, and closer to \$23.00 USD by the end of the continual study when the LLM is fairly good at the game. On average a human-bot game takes under 2 minutes. We pay workers \$0.06 USD per turn for human evaluation surveys, or \$0.08 USD if the turn annotation involves error modes. The estimated hourly wage is \$16.00 USD for human evaluation surveys. On average it takes under 2.5 minutes to annotate one game. We set the payment scheme through pilot studies and aim for a \$15.00 USD hourly wage.

Interface and Serving We implement MULTIREF using Empirica (Almaatouq et al., 2021) and on top of the code base of Gul and Artzi (2024). The speaker has 25 seconds to type into a chat box each turn and hit Enter or submit, and the listener has 45 seconds to click on the tangrams to select or to deselect. The game ends if one party idles for one turn, and the party idling is not compensated. We serve on an EC2 instance. We serve LLM policies with the Ray framework (Moritz et al., 2018). We walk through the first turns of a sample interaction in Figure 8, Figure 9, Figure 10, and Figure 11.

A.4 Artifacts Used and Data Release

Licenses The scientific artifacts we used, including the IDEFICS2-8B model (Laurençon et al., 2024) and the model serving framework Ray (Moritz et al., 2018), have open licenses (the Apache 2.0 License). We will release our model checkpoints and human-model interactions under Apache 2.0 as well.

Privacy Precaution We store encrypted MTurk worker ID via md5 hashing in our database. No other personal identifiable information is collected or released.

B Learning Details

B.1 Interaction Representation

We encode the context x as in Figure 12. We standardize action representation by ordering actions, for example, always produce `Select A C` rather than `Select C A`. We shuffle the context images during training as the order of context tangrams should not have any impact on the interaction logic.

B.2 Policy Initialization

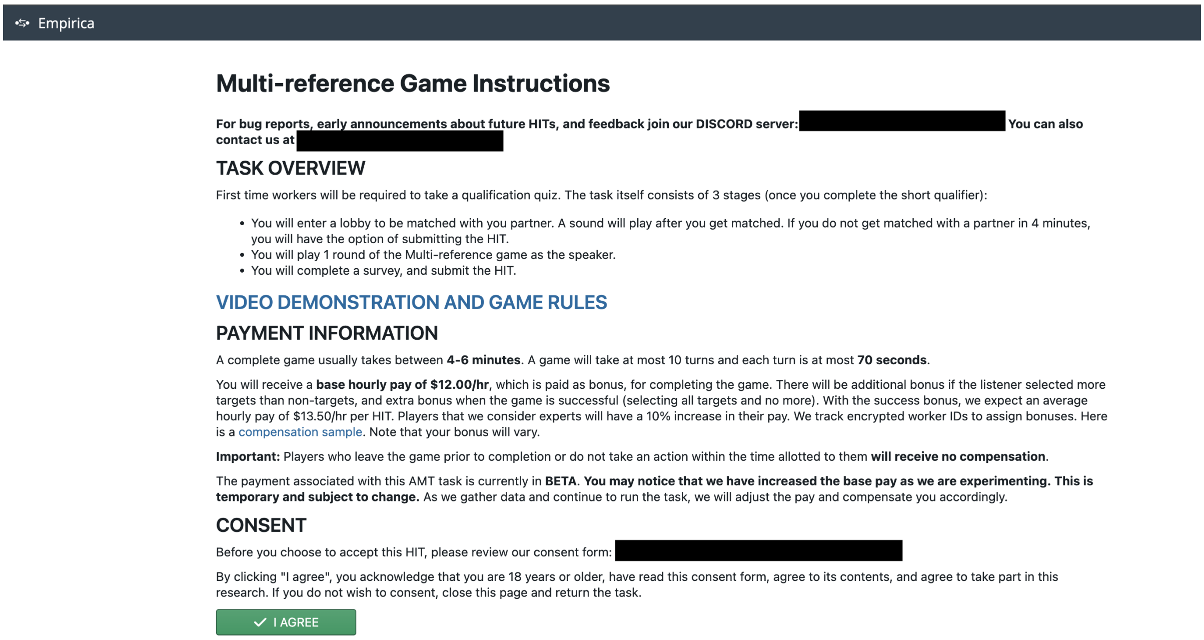
We seed the initial policy π_0 by fine-tuning the model on a small dataset of 90 successful turns D_0, where both the speaker and the listener are humans. We also experiment with prompting to initialize the policy. We find early that few-shot prompting yields a random policy at best, likely because reasoning with abstract shapes such as tangrams is visually out-of-distribution for the model.

There is a significant distribution shift between human-human interactions, and human-policy interactions, especially early on when the model performs poorly. In practice, two major differences are the length of interactions and the prevalence of deselection instructions, which are rare in human-human interactions. We address the deselection issue with data augmentation. We synthetically generate turns where the speaker asks for deselections, and the listener complies. We augment the data with these at a ratio of 1:12 to the existing data. This helps the LLM policy learn to deselect and recover from mistakes. This augmentation is only used for D_0 and such distribution shift is not present in later rounds, when learning from actual human-bot interactions.

We validate our design online with 30 main-split human-bot pilot interactions, or offline with a validation set of 344 successful main-split human-human turns.

B.3 Hyperparameters and Other Implementation Details

We use the instruction-tuned IDEFICS2-8B model for all policies. We fine-tune with LoRA adapters (Hu et al., 2022) (α=r=8, dropout=0.1) due to compute constraints. Appendix D provides more LoRA details. We train each model with a single GPU, RTX A6000, NVIDIA A100 40GB or 80GB. The time to train ranges between 2–24 hours, longer in later rounds as more data accumulates. For stopping criteria, we pick checkpoints by highest accuracy (exact match) among three seeds


Empirica

Multi-reference Game Instructions

For bug reports, early announcements about future HITs, and feedback join our DISCORD server: [redacted] **You can also contact us at** [redacted]

TASK OVERVIEW

First time workers will be required to take a qualification quiz. The task itself consists of 3 stages (once you complete the short qualifier):

- You will enter a lobby to be matched with you partner. A sound will play after you get matched. If you do not get matched with a partner in 4 minutes, you will have the option of submitting the HIT.
- You will play 1 round of the Multi-reference game as the speaker.
- You will complete a survey, and submit the HIT.

VIDEO DEMONSTRATION AND GAME RULES

PAYMENT INFORMATION

A complete game usually takes between **4-6 minutes**. A game will take at most 10 turns and each turn is at most **70 seconds**.

You will receive a **base hourly pay of $12.00/hr**, which is paid as bonus, for completing the game. There will be additional bonus if the listener selected more targets than non-targets, and extra bonus when the game is successful (selecting all targets and no more). With the success bonus, we expect an average hourly pay of $13.50/hr per HIT. Players that we consider experts will have a 10% increase in their pay. We track encrypted worker IDs to assign bonuses. Here is a compensation sample. Note that your bonus will vary.

Important: Players who leave the game prior to completion or do not take an action within the time allotted to them **will receive no compensation**.

The payment associated with this AMT task is currently in **BETA**. **You may notice that we have increased the base pay as we are experimenting. This is temporary and subject to change.** As we gather data and continue to run the task, we will adjust the pay and compensate you accordingly.

CONSENT

Before you choose to accept this HIT, please review our consent form: [redacted]

By clicking "I agree", you acknowledge that you are 18 years or older, have read this consent form, agree to its contents, and agree to take part in this research. If you do not wish to consent, close this page and return the task.

I AGREE

Figure 8: The MULTIREF instruction page for workers including compensation details and game rules.

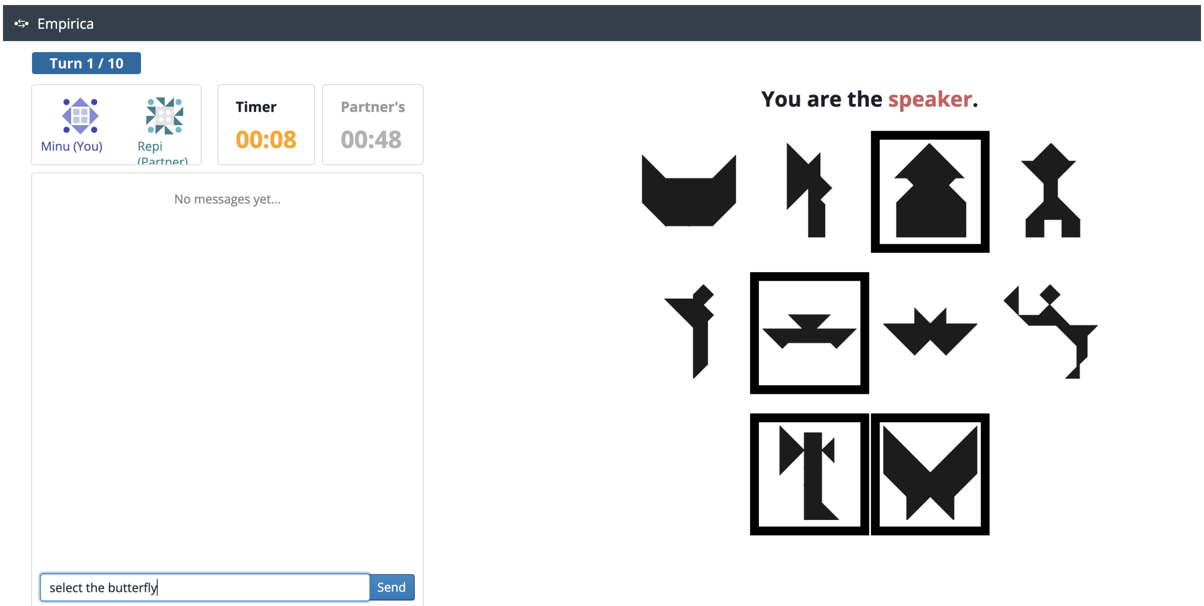


Figure 9: The MULTIREF interface for the speaker in turn 1. Predefined targets are revealed to the speaker in black boxes.

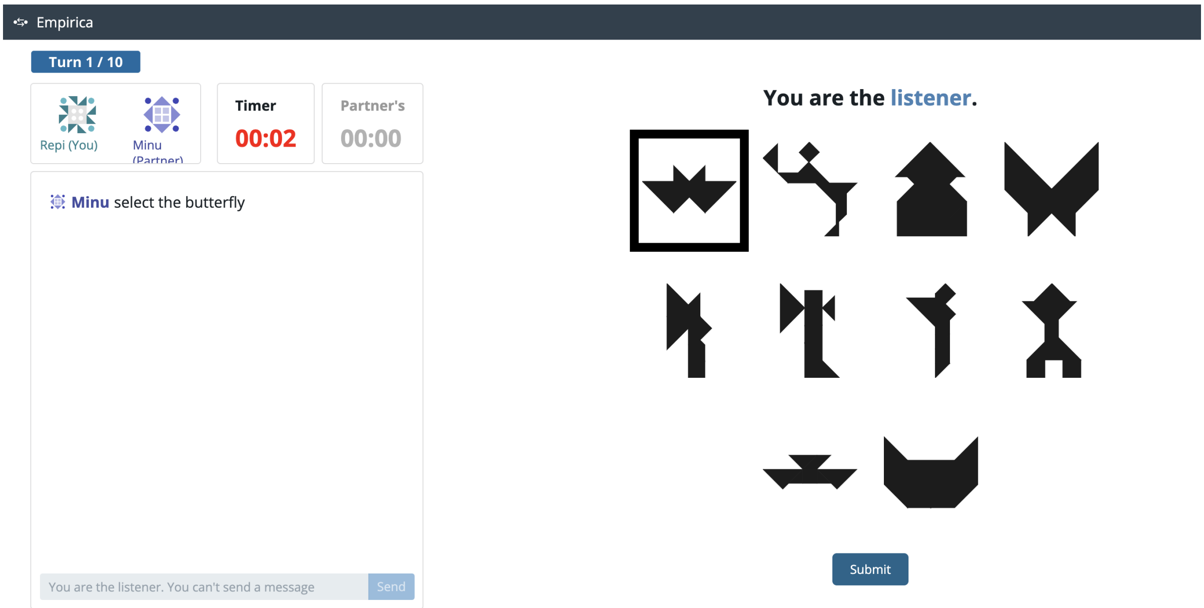


Figure 10: The MULTIREF interface for the listener in turn 2, following the speaker turn in Figure 9. Targets are hidden for the listener, and the context tangrams are in a different order. Here the listener has selected a tangram given the instruction *select the butterfly*.

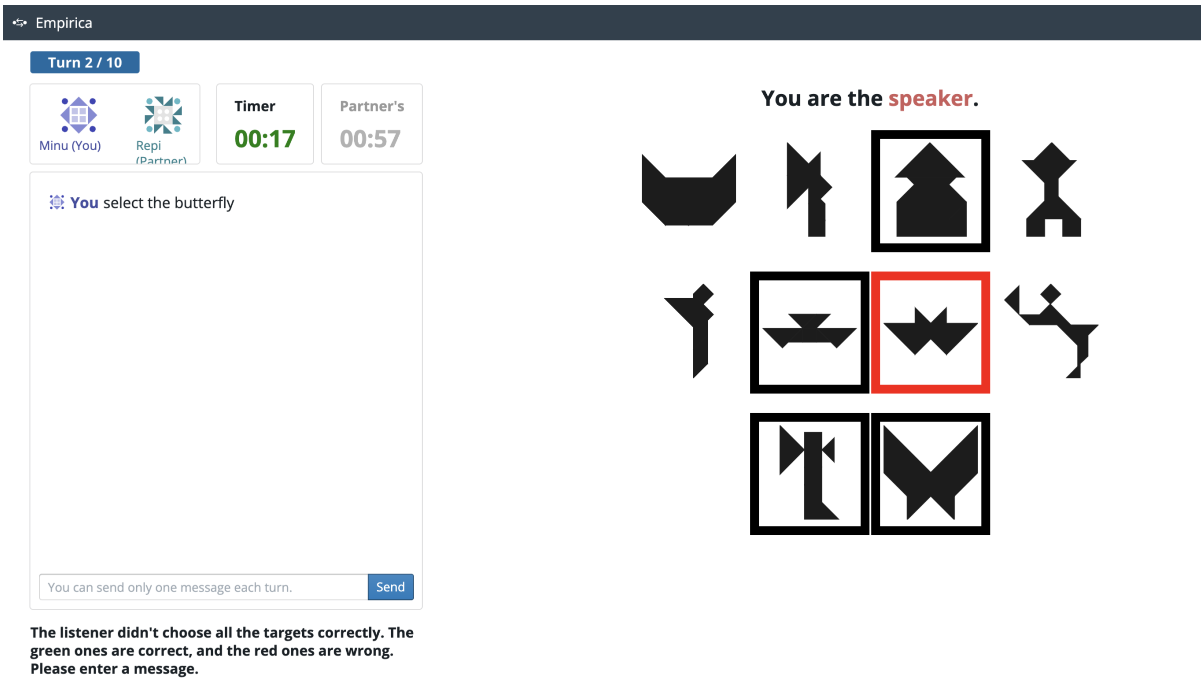


Figure 11: The MULTIREF interface for the speaker in turn 3, following the listener turn in Figure 10 The listener selected a non-target tangram, shown in red to the speaker.

System: <img>A <img>B <img>C <img>D <img>E <img>F <img>G <img>H <img>I <img>J ..... *(Context tangrams tagged by code names)*
System: none is selected .... *(Turn 1 state)*
User: Select the letter S with the large triangle back side .... *(Turn 1 utterance)*
Assistant: Select J .... *(Turn 1 action)*
System: J currently selected
User: unselect that one and try again
Assistant: Deselect J select A
System: A currently selected
User: almost, unselect that one and try one more time
Assistant: Deselect A select G
System: G currently selected
User: correct. Now select the rabbit facing right
Assistant: Select F
System: F G currently selected
User: no that is a bird, try again
Assistant: Deselect F select H
System: G H currently selected
User: no that is another bird, try again
Assistant: **Deselect H select F** .... *(Action)*

Figure 12: Policy prompt example with a model predicted **action** and additional *comments* for readability.

[Previous turns omitted]
System: none is selected ..... *(Previous turns)*
Speaker: Man in a hat
Listener: Select A
System: A currently selected *(Augmented state)*
Speaker: Wrong, undo what you selected *(Augmented utterance)*
Listener: **Deselect A** ..... *(Augmented action)*

Figure 13: An example of deselection augmentation with augmented **action** and *comments*.

on a held-out validation set of 344 turns $D^{\text{HH}}_{\text{val}}$. The validation set is curated from 92 human-human games in the main split of tangrams. We summarize hyperparameters in Table 1.

Data Imbalance The decoded feedback is imbalanced, with more negative examples than positive examples (3:1 to 2:1), especially at early rounds of continual learning. We address this by weighting the loss by the absolute value of the reward, i.e., -0.1 for RL or λ_d and λ_u for KTO, and by downsampling negative examples per batch, such that the number of positive examples and negative examples is roughly 5:4.

KTO Stability Deviation from the original KTO implementation by higher learning rate, higher β, more epochs, produces better results empirically on the validation set in pilot and round $\rho = 1$. However, in round $\rho = 2$, B-KTO policy starts to degenerate by producing nonsensical actions such as `Deselect A select A B` or `Deselect select select`. We attempt to mitigate this issue during training round $\rho = 3$ by switching from weighting $\lambda_d = 4$ and $\lambda_u = 1$ as recommended in Ethayarajh et al. (2024) to $\lambda_d = \lambda_u = 1$, plus downsampling negative examples. We also introduce regex-based constrained decoding to prevent nonsensical actions for B-KTO and T-KTO policies in round $\rho = 3$. Despite that, the KTO group performs worse in live interactions (Figure 4). We suspect KTO is more challenging to optimize for iterative continual learning, but we suspect further tuning (with higher computational costs) can reduce or even eliminate these issues.

B.4 Evaluation Metrics

Interaction-level Metrics Interaction performance and statistics are computed automatically from live deployment interactions. They do not require further annotation.

1. **Success rate** = # successful interactions / # all interactions. An interaction is successful if the listener selects all and only targets before running out of 10 turns. This is the primary metric we use to evaluate the performance of the LLM policy.

2. **# Turns per interaction**. This is a measure of collaborative efficiency.

Turn-level Metrics with Reference to Human Annotation We compute turn-level metrics either with respect to HH games where we consider

Hyperparameter	Search Space	FFT	RL	KTO
Optimizer		AdamW	AdamW	RMSProp
Learning rate	{1e-6, 1e-5, 1e-4, 2e-4}	1e-4	1e-4	1e-5
Learning rate decay	{no, cosine, linear}	cosine	cosine	no
Epochs	{5, 10, 20, 40}	20	20	20
Warm-up steps	{0, 10, 50}	10	10	10
Weight decay	{0, 0.01, 0.1}	0.01	0.01	0.01
Effective batch size	{16, 32, 48, 64, 128}	64	64	64
Entropy weight	{0, 0.01, 0.5, 0.1}	0.01	0.01	0.1
β_{KTO}	{0.01, 0.1, 0.5}			0.5
Temperature		1	1	1

Table 1: Hyperparameter settings.

human listener action as ground truth (e.g., $D^{\text{HH}}_{\text{val}}$), or with respect to B-FFT games where we consider actions a^* annotated in post-hoc surveys as ground truth. When computed with live interactions, these metrics are biased towards longer or failed interactions because they have more turns than successful interactions.

1. **Exact match** = # exact match / # all turns. An exact match is when the action taken by the policy matches exactly the action labeled/taken by human listeners ($\hat{a} = a^*$).

2. **Similarity** = Sim$(\hat{a}, a^*)$ is a composite metric. Let $f(p, q) : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$ be a function that evaluates the similarity between two images $p, q \in \mathcal{I}$. Let the action taken by policy be $\hat{a} = \{\hat{p}_1, \hat{p}_2, ..., \hat{p}_{\hat{n}}, \hat{q}_1, \hat{q}_2, ..., \hat{q}_{\hat{m}}\}$ where p are the selected tangrams and q are the deselected tangrams. Denote the ground truth actions as $a^* = \{p^*_1, p^*_2, ..., p^*_{n^*}, q^*_1, q^*_2, ..., q^*_{m^*}\}$. The similarity between two actions is defined as:

$$\text{Sim}(\hat{a}, a^*) = \frac{1}{\hat{n}n^* + \hat{m}m^*}\left(\sum_{i=1}^{\hat{n}}\sum_{j=1}^{n^*} f(\hat{p}_i, p^*_j) + \sum_{i=1}^{\hat{m}}\sum_{j=1}^{m^*} f(\hat{q}_i, q^*_j)\right). \quad (3)$$

If only one of $\hat{n}$ and n^* is zero, we rewrite $\Sigma_{i=1}^{\hat{n}}\Sigma_{j=1}^{n^*} f(\hat{p}_i, p^*_j)$ with $-\max(\hat{n}, n^*)$, and $\hat{n}n^*$ in the denominator with $\max(\hat{n}, n^*)$, intuitively assigning -1 for each missed selection. This edge case is similarly treated for $\hat{m}$, m^* and deselection. We compute similarities using embeddings from the tangram fine-tuned CLIP model of Ji et al. (2022) .

3. **Positive feedback** = # turns receiving positive feedback / # all turns. An action receives positive feedback if speaker is satisfied with the listener's action in the follow-up interaction. This is labeled in human evaluation survey.

Intuitively, a click is approximately accurate if it selects a target or deselects a non-target. We compute this for all clicks from all interactions in a round for all systems in Figure 4.

Corpus-level Metrics We analyze speaker instructions per system-round. The keywords used to generate the analysis in Figure 7 are:

1. **# Reset** = occurrences of phrases in {*reset, restart, from scratch, all over, start over, deselect everything, deselect all, remove everything, remove all, clear everything, clear all, unselect everything, unselect all, drop everything, drop all* }

2. **# Try again** = occurrences of phrases in {*try again, try one more time, the other one* }

C Cumulative Number of Interactions Observed

The main text includes results by round. We collect roughly 330 interactions per policy per round. Due to the uncertainty of live data collection, we do not always hit this exact number for each variant and round. Figure 14 shows the cumulative number of human-bot interaction seen by a policy variant in each round.

D Additional Enhanced LoRA Launch

We suspect the plateau of B-FFT in Figure 4 is partially due to the limited expressivity of LoRA

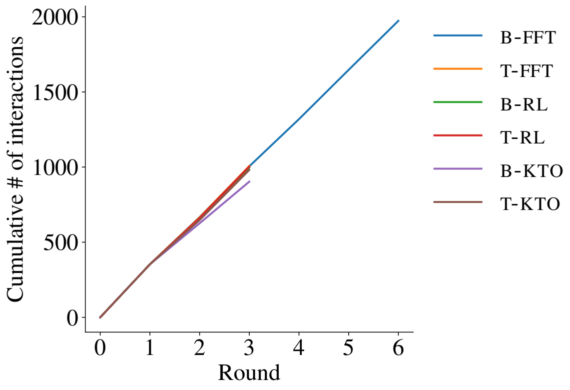


Figure 14: Cumulative number of human-bot interactions used to train the policy each round.

adapters we used. We test this hypothesis by deploying round $\rho = 4$ and $\rho = 5$ again with enhanced LoRA adapters. We use the same hyperparameters as in Section B.3 except for the additional adapters. The original adapter placement is on the text model, the modality projector, and the perceiver resampler. Adapters include the down projection layers, the gate projection layers, the up projection layers, and the key/query/value projection layers. In comparison, the enhanced launch adds adapters on the vision model, including the out projection, the first and the second fully connected layers, besides the projection layers on text models. Figure 15 shows the results from this complementary deployment. The enhanced LoRA adapters yield a small improvement in interaction success rate compared to the original launch, yet the overall slowdown is evident. This suggests LoRA expressivity has some effect, but other effects are also limiting the LLM policy from continuing its earlier improvement trends.

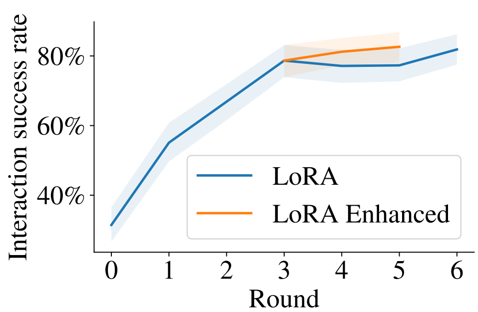


Figure 15: Success rate of B-FFT with additional LoRA adapters in round 4 and 5.

Round	0	1	2	3	4	5	6
B-FFT	31.4	55.1	**66.9**	**78.7**	77.1	77.3	81.9
T-FFT	31.4	55.8	67.8	74.0	-	-	-
B-RL	31.4	50.0	64.4	70.7	-	-	-
T-RL	31.4	**56.8**	62.4	70.3	-	-	-
B-KTO	31.4	45.1	52.0	46.9	-	-	-
T-KTO	31.4	50.0	61.7	66.1	-	-	-
CONTROL	31.4	-	-	-	-	-	33.0
HH	-	-	-	-	-	-	100.0

Table 2: Interaction task success rate in percentage (↑). We collect roughly 330 human-bot games per datapoint, except for HH where we only collect 50 games. Round 0 is shared among systems, except for HH. All system are deployed for three rounds, and the top performing one (B-FFT) is deployed for additional three rounds; preempted or not-applicable rounds are marked with dash (-). We **bold** the highest task success rate in a round.

Round	0	1	2	3	4	5	6
B-FFT	8.87	8.16	**7.33**	**6.99**	6.92	6.87	6.71
T-FFT	8.87	7.95	7.44	7.14	-	-	-
B-RL	8.87	8.10	7.42	7.22	-	-	-
T-RL	8.87	**7.94**	7.60	7.24	-	-	-
B-KTO	8.87	8.15	7.66	8.03	-	-	-
T-KTO	8.87	8.06	7.56	7.27	-	-	-
CONTROL	8.87	-	-	-	-	-	8.75
HH	-	-	-	-	-	-	4.61

Table 3: # turns per interaction (↓). Maximum 10 turns. Each game has 3-5 targets and HH games usually take one turn per target. We **bold** the fewest # turns per interaction in a round.

E Detailed Results

We present numerical results of metrics for interaction level performance in Figure 2 (Table 2, Table 3, Table 4), human evaluation performance in Figure 5 (Table 5, and language analysis in Figure 7 (Table 6, Table 7, Table 8, Table 9).

F Feedback Decoder Details

Design The prompt design is minimal, general, and task-agnostic. We validate the prompt with manual inspection prior to continual learning launch and human surveys. Considering only the most recent two action-utterance turns $\langle \hat{a}_{i-1}, u_i, \hat{a}_i, u_{i+1} \rangle$ is sufficient to produce satisfactory decoding results, and more history seems to distract the decoder. We also experimented with numerical reward (i.e., decoding a real number), experimenting with a discretized reward space of $\{.0, .1, .5, .9\}$. Our experiments show the model is not well calibrated for such decoding.

Round	0	1	2	3	4	5	6
B-FFT	59.7	64.0	**67.2**	**69.9**	69.8	69.5	72.2
T-FFT	59.7	**65.2**	67.1	68.9	-	-	-
B-RL	59.7	63.8	66.6	68.8	-	-	-
T-RL	59.7	64.9	65.0	67.0	-	-	-
B-KTO	59.7	60.7	61.6	58.5	-	-	-
T-KTO	59.7	62.1	63.2	64.0	-	-	-
CONTROL	59.7	-	-	-	-	-	60.5
HH	-	-	-	-	-	-	89.3

Table 4: Click accuracy in percentage (↑). We **bold** the highest click accuracy in a round.

Round	0	1	2	3	4	5	6	CONTROL	HH
Exact match	30.7	38.4	44.8	47.2	48.7	46.7	52.3	31.7	79.1
Pos Feedback	33.0	39.2	43.1	47.5	49.1	49.4	50.4	34.6	78.4
Sim($\hat{a}, a^*$)	19.0	34.8	42.5	46.0	47.7	43.5	51.3	19.4	83.8
Sim($\hat{a}, a^*$) -FB	0.0	13.6	19.2	23.9	23.3	15.6	25.7	1.4	67.9

Table 5: Turn level performance of B-FFT based on human evaluation, all in percentages (↑).

Round	0	1	2	3	4	5	6
B-FFT	1458	1400	1283	1179	1206	1211	1307
T-FFT	1458	1361	1279	1206	-	-	-
B-RL	1458	1352	1248	1187	-	-	-
T-RL	1458	1310	1306	1164	-	-	-
B-KTO	1458	1324	1183	1238	-	-	-
T-KTO	1458	1303	1332	1184	-	-	-
CONTROL	1458	-	-	-	-	-	1311
HH	-	-	-	-	-	-	433

Table 6: Vocabulary size of speaker instructions when humans interact with different systems across rounds.

Round	0	1	2	3	4	5	6
B-FFT	8.78	8.87	7.94	8.49	8.53	8.30	9.23
T-FFT	8.78	8.69	8.24	8.49	-	-	-
B-RL	8.78	8.29	7.94	8.45	-	-	-
T-RL	8.78	8.42	8.39	8.26	-	-	-
B-KTO	8.78	8.29	8.59	8.57	-	-	-
T-KTO	8.78	8.05	8.41	8.05	-	-	-
CONTROL	8.78	-	-	-	-	-	8.19
HH	-	-	-	-	-	-	8.49

Table 7: Utterance length of speaker instructions when humans interact with different systems across rounds.

Round	0	1	2	3	4	5	6
B-FFT	19	11	14	7	6	9	6
T-FFT	19	5	3	2	-	-	-
B-RL	19	10	17	9	-	-	-
T-RL	19	3	9	6	-	-	-
B-KTO	19	17	42	47	-	-	-
T-KTO	19	13	21	8	-	-	-

Table 8: # Reset words of different systems across rounds.

Round	0	1	2	3	4	5	6
B-FFT	42	24	14	3	4	4	14
T-FFT	42	15	7	5	-	-	-
B-RL	42	16	12	4	-	-	-
T-RL	42	21	15	6	-	-	-
B-KTO	42	20	5	9	-	-	-
T-KTO	42	11	6	6	-	-	-

Table 9: # Try again words of different systems across rounds.

User: Please carefully read the following conversation and answer: Is the very last utterance from the speaker positive or negative feedback? Often negative feedback include corrections and keywords like no, not, undo, don't, with generally negative sentiment, while positive feedback often includes good, yes, correct, okay, or simply move on to the next stage. Lean towards negative if it sounds neutral.
(start of the conversation)
Speaker: house
Listener: Select F .... *(Action to focus on)*
Speaker: horned roof .... *(Feedback)*
(end of the conversation)
Answer a single word, Positive, or Negative
Assistant: **Negative**

Figure 16: Feedback decoder false-negative example: the feedback decoder fails to recognize an implicit positive feedback from the speaker by moving on to the next target. The verbal **feedback generated by the model** is in bold. Additional *comments for readability* are in italics.

Feedback Decoder Error and Potential Fix Roughly 15% of feedback decoder predictions are false negatives, see Figure 6 top row, and an example in Figure 16. We handle negatives in different ways in our experiments, but generally negatives examples have less impact than positive ones, so the learner is robust to false negative noise. Of course, it does mean that we are losing valuable positive data, and reducing this error rate is an important direction for future work. This can potentially speed up learning further.

G Use of AI Assistant

Copilot is used for code generation assistance. ChatGPT is used for formatting in LaTeX.

H Interaction Case Studies

Figures 17–20 illustrate the diversity of MULTIREF interaction scenarios. Black borders indicate targets. Yellow dots indicate actions taken by the listener. Green borders indicate correct selections, while red borders indicate wrong selection.

We also fix a game context and compare the behavior of the initial π_{θ_0} (failed at the game, Figure 21), the final π_{θ_6} (succeeded at the game within 6 turns, Figure 22), and a human listener (succeeded within 2 turns, Figure 23).

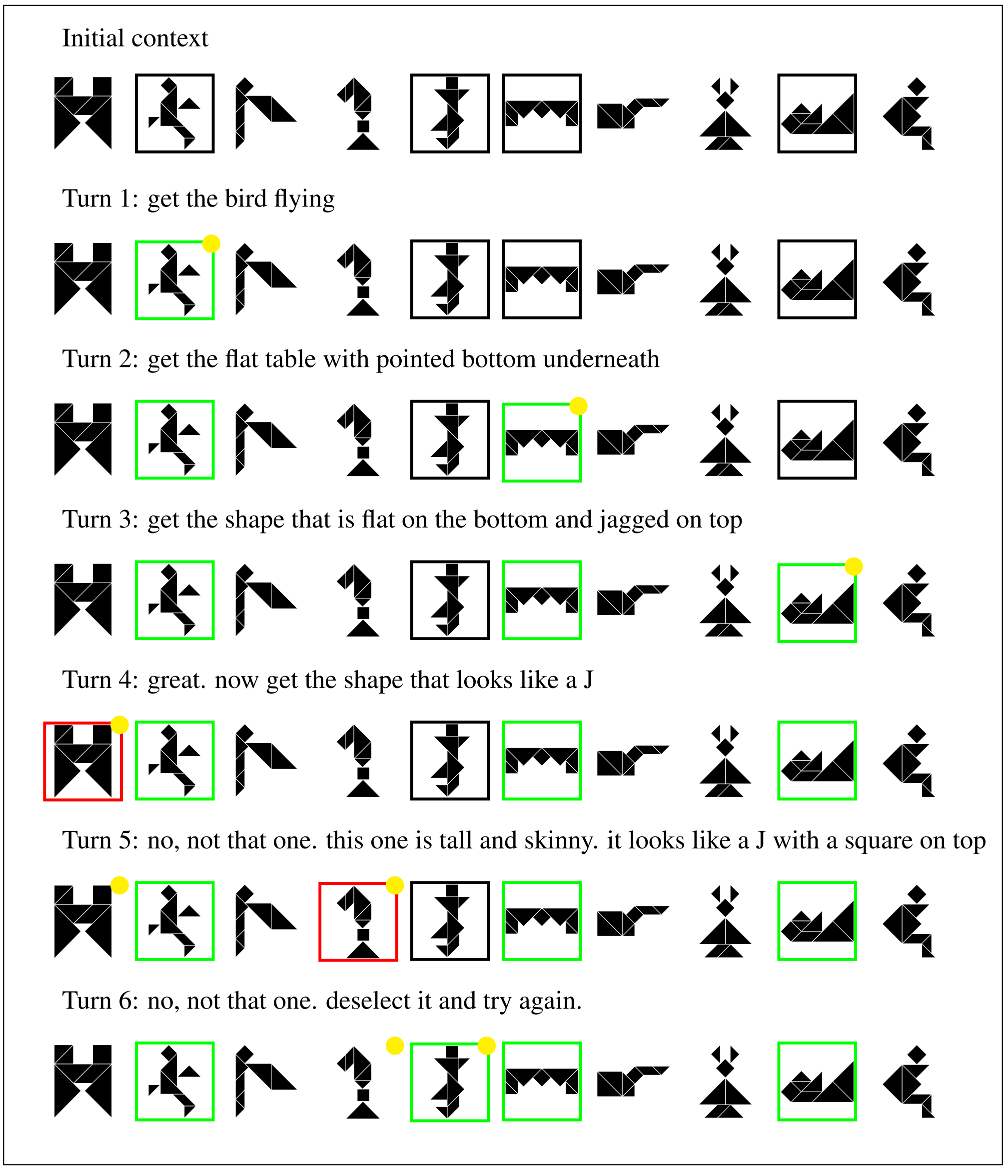


Figure 17: The speaker is left with the last target at Turn 4. Failing, they provide an additional description in Turn 5, and eventually resort to “try again" without describing the target in Turn 6. The initial turns illustrate how feedback is implied, rather than specified explicitly. The interaction concludes successfully.

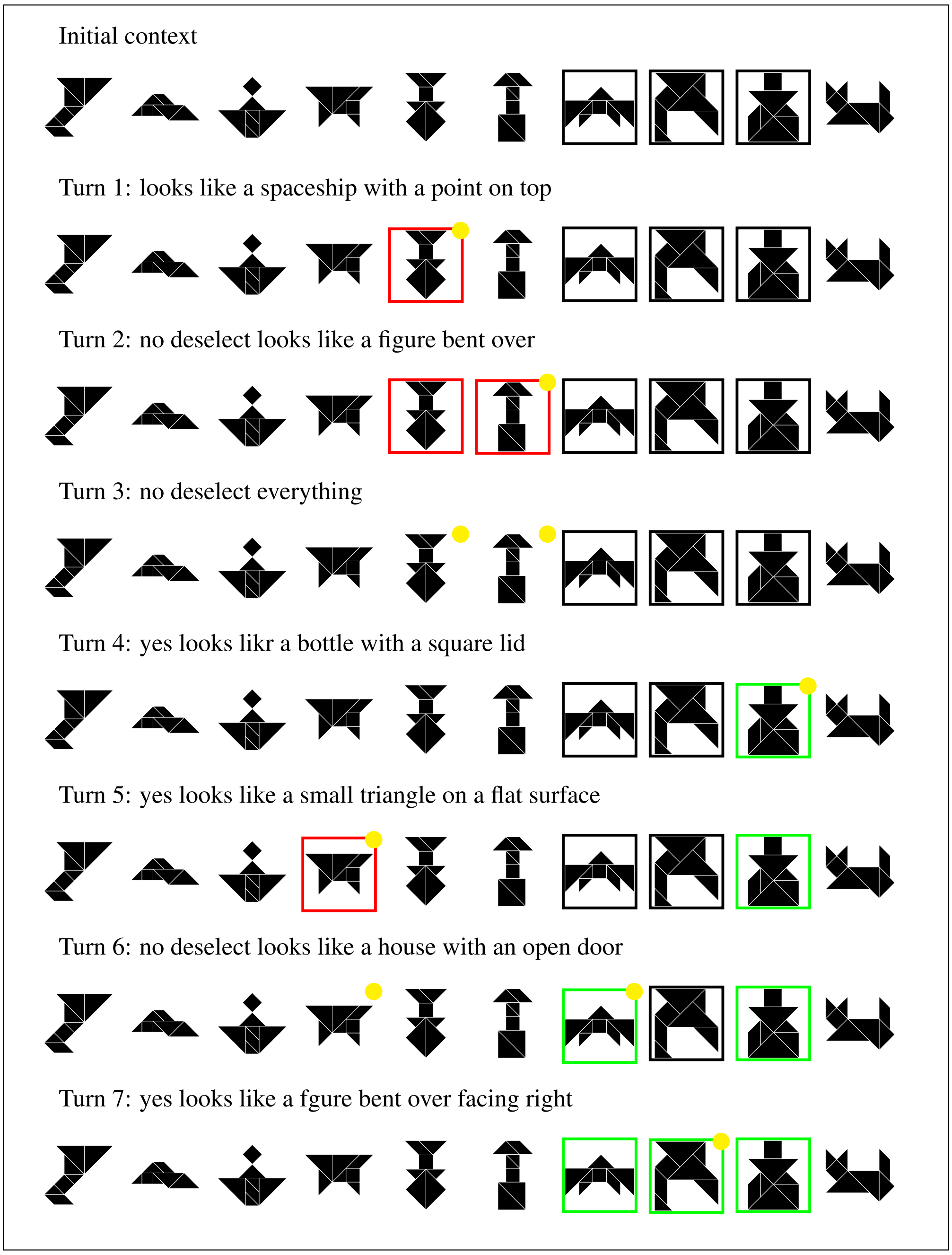


Figure 18: The speaker asks to deselect everything in Turn 3 to reset, an expression of frustration. The interaction concludes successfully.

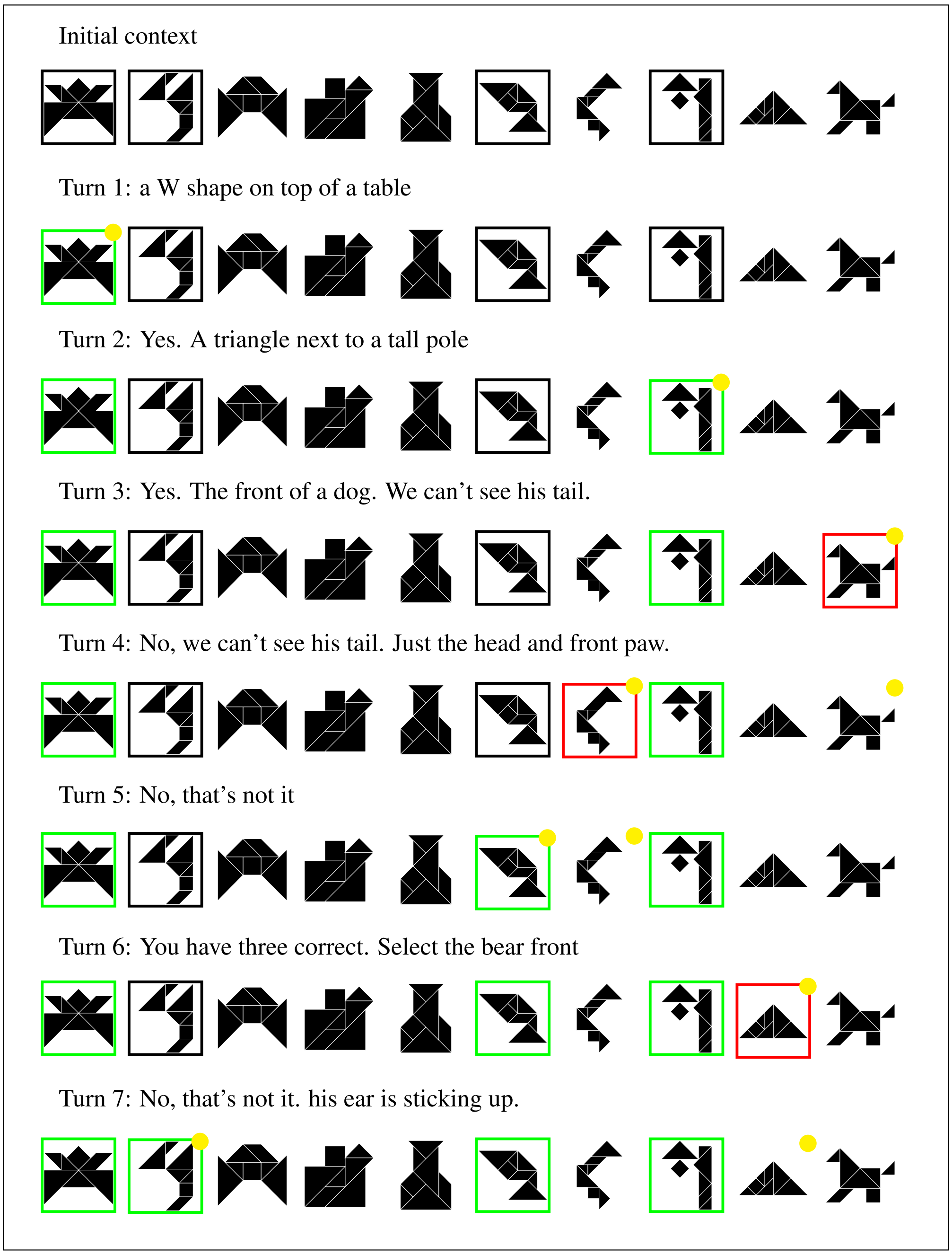


Figure 19: The abstractness and ambiguity of tangrams lend to complex interactions. There are two dogs in the context, and the listener struggles to disambiguate or identify the target. The interaction concludes successfully.

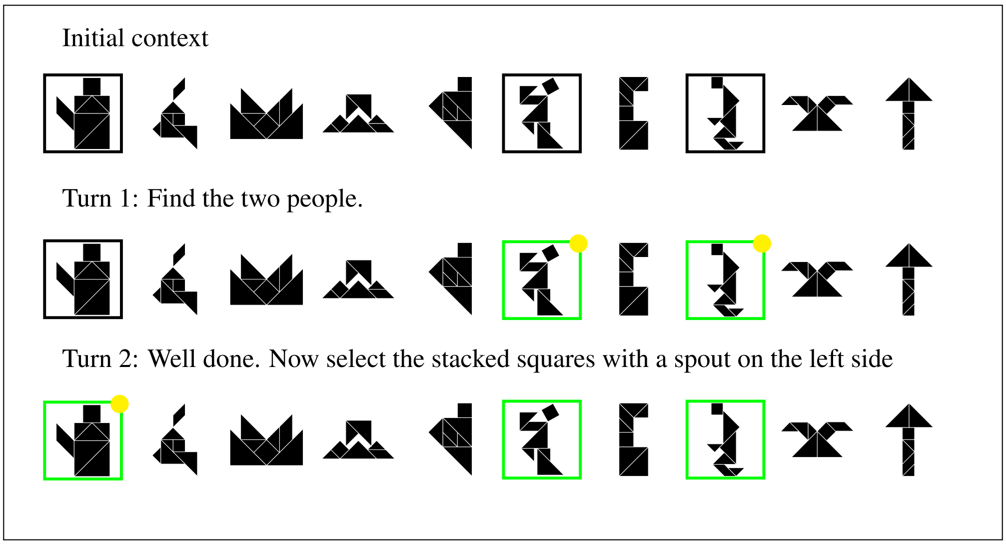


Figure 20: The speaker asks for two targets in Turn 1, exemplifying Grice's Maxims of Quantity - one tries to be as informative as one possibly can, and gives as much information as is needed, and no more (Grice, 1975). The interaction concludes successfully.

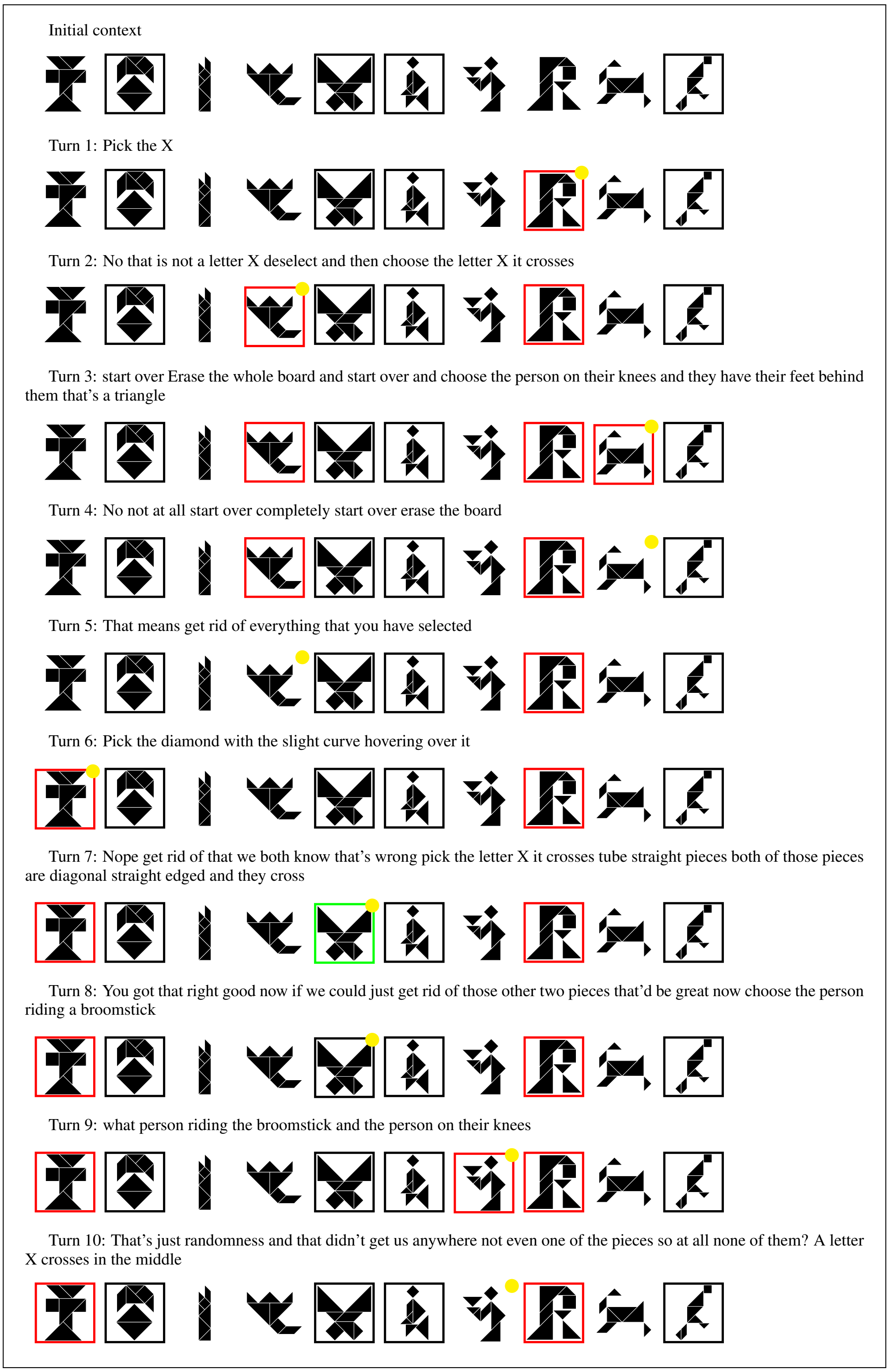


Figure 21: An interaction trace between a human speaker and the initial listener policy π_{θ_0}. This interaction concludes unsuccessfully.

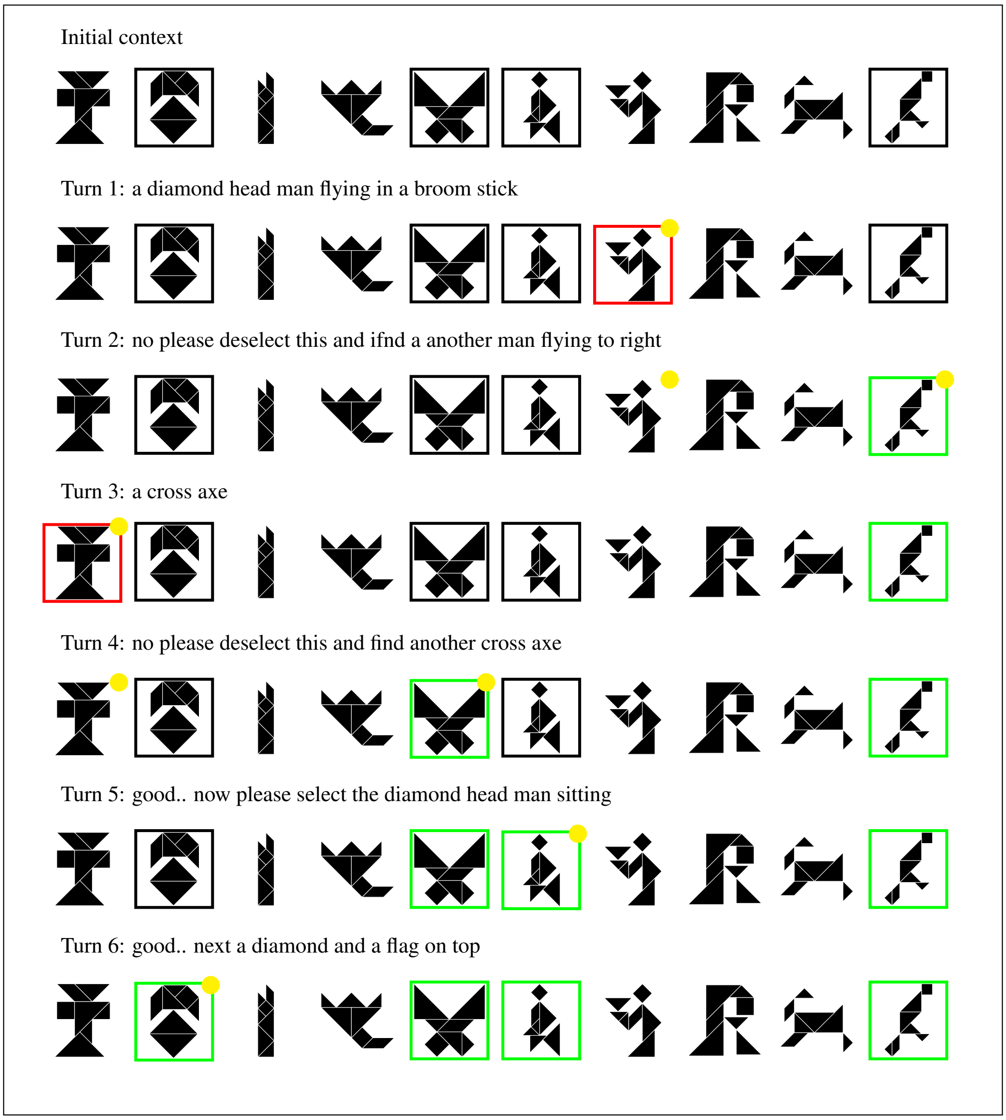


Figure 22: An interaction trace between a human speaker and the final listener policy π_{θ_6}. This interaction concludes successfully.

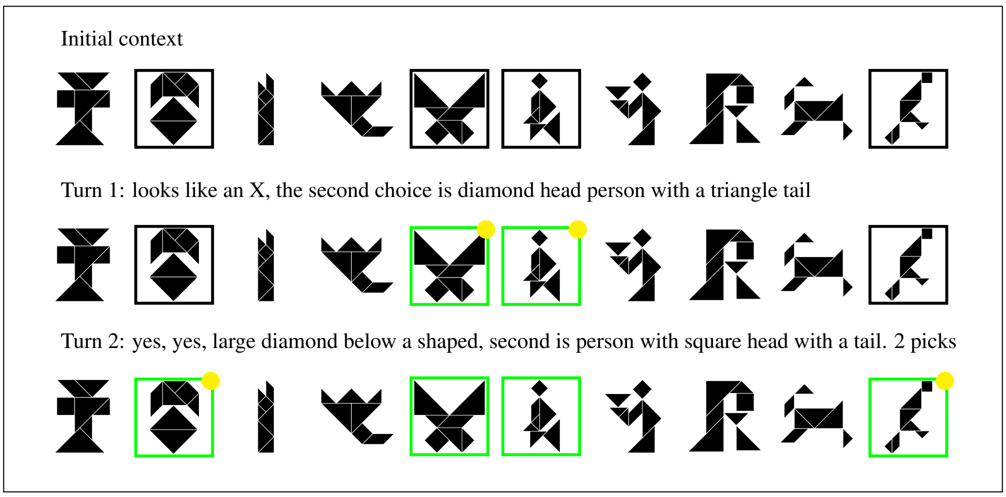


Figure 23: An interaction trace between a human speaker and a human listener. This interaction concludes successfully, even faster than Figure 22