1 Behavioral Experiment Details

In this section, we provide additional details about our experiment.

1.1 Task User Interface

The full teacher interface during each episode is shown in Fig. 1. During the "action" phase, the teacher could watch the learner move but could not send messages. After the player acted (Fig. 2) the chat interface was enabled and the teacher could send feedback in the form of unlimited chat messages to the learner. The learner's interface was identical, except that (1) they only saw the right half of the display (e.g. shapes and colors, but not true object values), and (2) they received chat messages, but could not reply (e.g., they could not ask for clarification).

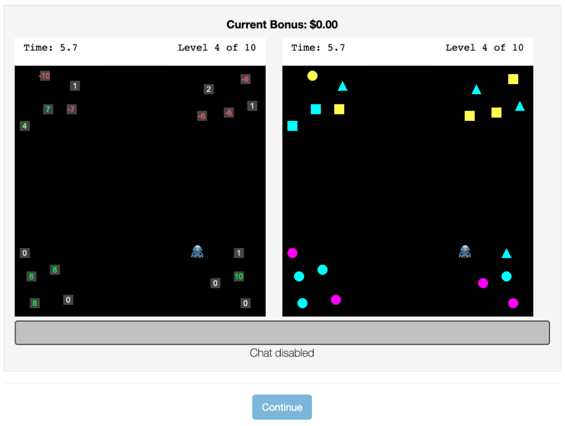
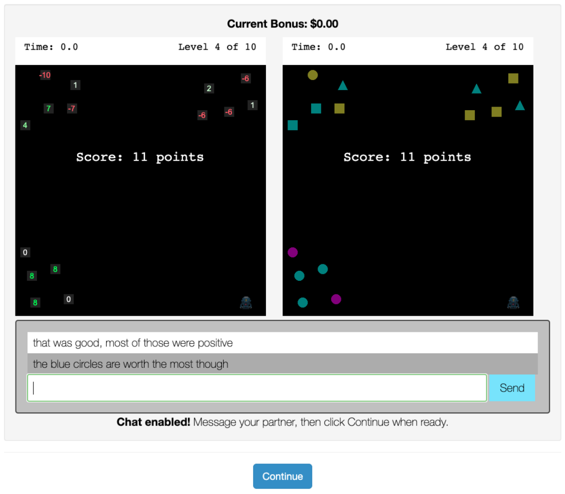


Figure 1: The teacher user interface during the "action" phase of each episode. The chat interface was disabled, so they watched the learner play the new level but could not send messages. This meant that they could not provide proactive specific advice (such as which corner to go to), only retrospective feedback.



that was good, most of those were positive

the blue circles are worth the most though

Chat enabled! Message your partner, then click Continue when ready.

Figure 2: The teacher user interface during the "feedback" phase of each episode. The chat interface was enabled, and they could send unconstrained messages to the learner.

1.2 Worker Qualifications

We required participants to be located in the United States and have an approval percentage > 95%. We did not allow repeat participants and used Google's reCAPTCHA v2 to filter out bot participants.

1.3 Reward Functions

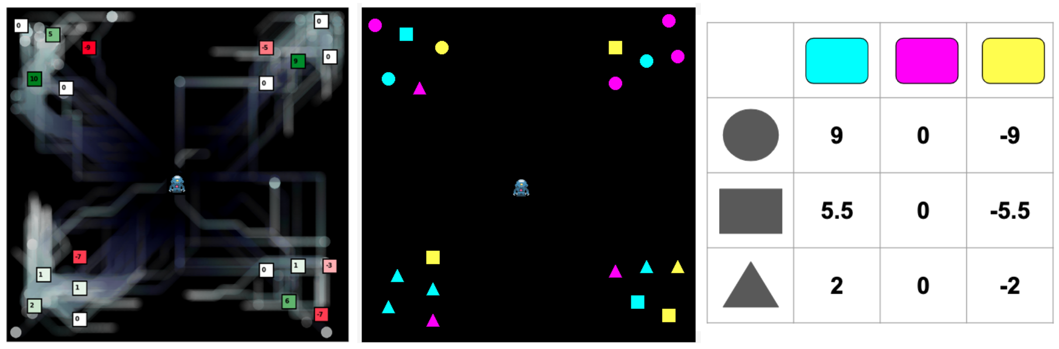
Fig. 3 shows the underlying object rewards, an example reward function, and the resulting learner user interface. Our full feature space consists of six base features (colors: yellow, blue, pink; shapes: square, circle, triangle) and their nine conjunctions (yellow|square, yellow|circle... etc). We designed the general structure of our reward functions to allow for multiple levels of learning: it is relatively easy to communicate that pink shapes are positive, but more nuanced that five pink squares are worth one pink circle. Because teachers communicated most about the more valuable shapes and colors, permuting the reward function (i.e., using all possible combinations as the most valuabe shape/color) ensured we collected language about all features.

1.4 Additional Text Analysis

Section 5 provides full transcripts from several pairs.

Curriculum Structure. As described in the main paper, we observed a natural curriculum structure in teachers' feedback. Teachers began with *descriptive* feedback, then switched to *evaluative*. Concretely, teachers told learners how to alter their behavior early on, then praised good behavior (or criticized poor behavior) in later episodes (see chat messages in Tables 4 and 5). It is perhaps useful to conceptualize this as two distinct phases. Teachers first provide "off-policy" descriptive feedback. Once the learner's feature-counts are generally aligned with the teacher's rewards, they provide "on-policy" positive feedback confirming their behavior is good. As noted in the main paper, Level 6 was challenging and players scored poorly (see Fig. 3), resulting in a shift *back* from "Evaluative" feedback to "Descriptive" or "Imperative" to correct the learner's actions.

Types of Feedback. The relatively low portion of *imperative* feedback can be attributed to our task structure: teachers could only provide feedback *after* the learner acted, so imperative feedback was limited to retrospective advice. If the teacher could issue *commands* prior to learner action, we would expect to see more imperative language. We also note that some feedback "blurs the lines" between types. For example, the imperative "GET VIOLET SQUARE" is actually "Descriptive" feedback since it is informative about the *features* of a preferred solution, rather than explicitly designating a *specific* preferred solution (which would take the form "get *that* violet square").



	🟦	🟪	🟨
●	9	0	-9
■	5.5	0	-5.5
▲	2	0	-2

Figure 3: Level 6. Left: Player actions (silver traces) and true object values, where green is positive and red is negative. Center: Player's user interface under one reward vector. Right: Reward vector used to generate this user interface. At this point, most learners knew which color was positive (in this case, blue). However, many learners did not understand shape valuations well enough to recognize the top left corner was worth more than the bottom left. These learners chose the three low-value positive objects (in this example, blue triangles) in the lower-left corner. This poor decision-making resulted in an uptick of "Descriptive" and "Imperative" feedback (see Figure 2C in main text).

When using descriptive language, teachers referenced colors (28%) more often than shapes (18%), which aligns with their relative importance in the reward function. Given unfettered communication, humans used a diverse set of language to express their preferences. Surprisingly few gave concrete, numerical feature information (e.g. "blue squares are worth 5 points"). Most gave relative values ("blue squares are better than triangles") or instructed learners to pursue particular features ("get blue squares"), as in Table 6, although Table 9 gives an example of a teacher who did give quantitative underlying rewards.

Sentiment. Teacher sentiment correlated with player score (Figure 4). This is largely due to increased positive evaluative feedback (praise) when the learner performed well. VADER provides a scalar sentiment in the range $\zeta \in [-1, 1]$. However, the reward associated with individual features in our task ranges $[-10, 10]$ and teachers often referenced multiple features. We thus applied a scaling factor to VADER's output (see Section 2.3). We also noted that the mode of the sentiment was 0. Many phrases simply name features without an explicit sentiment (for an example chat transcript demonstrating this, see Table 10). This motivated the addition of sentiment pragmatics (see Figure 5).

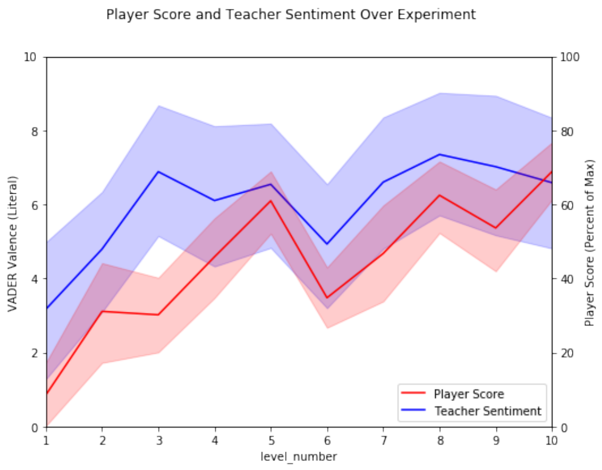


Figure 4: Player scores (red) and teacher sentiment (blue) over the 10 experimental trials. The two were correlated; notably, Level 6 was challenging resulting in a dip in player scores and teacher sentiment (see Fig. 3). Bands indicate 95% CI.

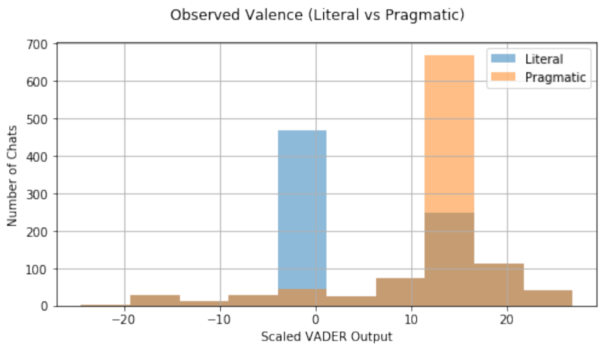


Figure 5: VADER sentiment output for chats in the experiment. Many teachers did not express an explicit sentiment in their chats, reflected in the spike at 0 for the "Literal" model. Observationally, these chats almost always carried *implied* positive sentiment; accordingly, we modified VADER to default to positive sentiment. This results in the rightward-shift in sentiment for "Pragmatic".

2 Bayesian Linear Regression Details

In this section, we provide additional details on our formulation and implementation of inverse reinforcement learning as Bayesian linear regression (sections 3, 5.1, and 5.2 of the main paper).

2.1 Bayesian Updates

Formally, on each utterance, we use **f** together with the sentiment ζ to perform Bayesian updates on the learner's prior beliefs:

$$\Sigma_{i+1} = \left(\Sigma_i^{-1} + \frac{\mathbf{f}\mathbf{f}^\top}{\sigma_\zeta^2}\right)^{-1} \tag{1}$$

$$\mu_{i+1} = \Sigma_{i+1}\left(\Sigma_i^{-1}\mu_i + \frac{\mathbf{f}\mathbf{f}^\top}{\sigma_\zeta^2} * \mathbf{f}\zeta\right) \tag{2}$$

This is illustrated in Fig. 6, which plots the learner's belief distribution $p(w_0), p(w_1), p(w_2)$ over a subset of features and multiple updates. The first piece of "Evaluative" feedback is somewhat ambiguous, as the model does not know which of its actions were valued by the teacher. A second piece of "Descriptive" feedback clarifies that the "Yellow Circle" feature was most likely the cause of the teacher's initial praise.

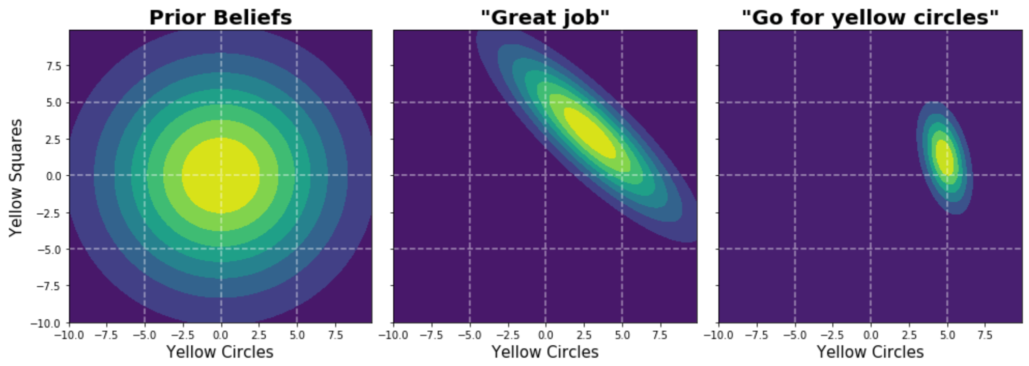


Figure 6: Probability isocontours over feature rewards through multiple updates. Left: Initialized model. Center: positive "Evaluative" feedback, using the trajectory from Figure 4 in the main paper (which collected all yellow shapes). This yields $\mathbf{f} = \|n_\phi(\tau)\|_1 = [.5, .5]$, e.g. weighting yellow circles and squares equally. The model "spreads credit" over both features. Right: We then provide positive "Descriptive" feedback for yellow circles (formally, $\mathbf{f} = [0, 1]$). The model settles on a high positive valuation for "Yellow Circles" and low positive for "Yellow Squares" as the most probable explanation for this pair of utterances.

2.2 Utterance Features

As described in Section 3.3 of the main paper, different feedback types relate to different elements of the learner's context. We use the trajectory shown in the main paper, Figure 4, to illustrate this in greater detail. This trajectory collected one yellow circle, one yellow square, and one yellow triangle. Table 1 shows actual feature vectors **f** resulting from different forms of feedback. Intuitively, "Descriptive" feedback provides more "focused" updates on specific features, while "Evaluative" and "Imperative" provide more "diffuse" information across many features.

2.3 Model Parameters

Key tunable parameters of the model are σ_ζ (the uncertainty associated with each observation) and the scaling factor for the sentiment (which is originally bounded $[-1, 1]$). We ran a grid search over

Utterance	Type	Yellow Circle	Yellow Square	Yellow Triangle	Pink Circle	Pink Square	Pink Triangle
"Great job"	Eval	.33	.33	.33	0	0	0
"Bottom left was better"	Imp	.4	.2	0	.2	.2	0
"Go for yellow circles"	Desc	1	0	0	0	0	0
"Pink triangles are best"	Desc	0	0	0	0	0	1

Table 1: Feature vectors **f** resulting from different utterances, assuming the example trajectory shown in Figure 4 of the main paper. "Evaluative" feedback informs the learner about the value of features obtained by their prior behavior; "Imperative" relates to alternative actions they *could* have taken, and "Descriptive" provides direct information about specific features.

these parameters and found that the model was relatively insensitive to different choices of σ_ζ but that increasing the sentiment scaling from 10 to 30 improved model performance dramatically.

Intuitively, this makes sense. First, we found VADER rarely registered a sentiment greater than .5 (see Fig. 5). Second, "Evaluative" and "Imperative" feedback often reference multiple features. This meant that a relatively "low" sentiment value could be applied to multiple objects. For example, if the learner collects one object worth 10 points and one worth 5 and the teacher says "Good job", the model should ideally register a sentiment of +15 to be distributed to the two. Scaling by 30 brings VADER's output closer to this. After choosing the literal model parameters, we re-ran the same analysis for parameters for the second "Pragmatic" update. The full results of our search are illustrated in Fig. 7.

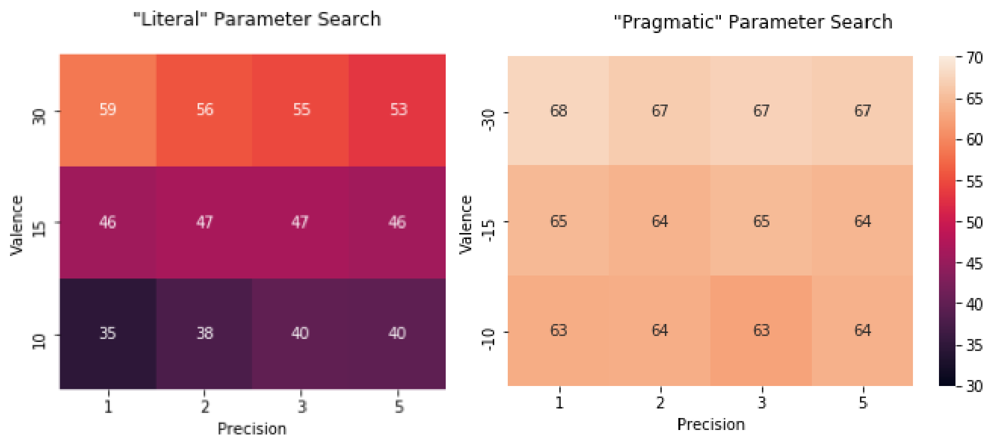


Figure 7: Left: "Literal" model parameter tuning. We plot mean score after 10 randomly sampled interactions, with a grid search over sentiment scaling (Y axis) and precision scaling (X axis). We found that the "Literal" model was sensitive to appropriate "Sentiment" scaling, but less sensitive to "Precision." We chose $\zeta = 30, P_\zeta = 2$ (equivalently, $\sigma_\zeta = .5$) because the greater precision reduced variance slightly. Right: "Pragmatic" update parameter tuning, with a "Literal" base model set to Sentiment=30 and Precision=2. We found the "Pragmatic" updates were relatively insensitive to parameters, and used the same parameters as the "Literal" update ($\zeta = -30, P_\zeta = 2$) for simplicity.

2.4 Grounding Function: Phrase Classifier

To implement our grounding function f_G, we ran a pilot experiment, labeled the resulting utterances, and trained a logistic regression. The pilot consisted of 81 pairs playing the same game, but using different colors (it included white) with correspondingly more complex reward functions. Two conditions were run: in one condition, the teacher could see the player's screen (as in the final experiment). In the second condition, the teacher could *not* see the player's screen; they could only see the object's location and values, and the player's behavior. This prevented them from using "Descriptive" feedback. Thus, the training data contains a more heterogenous set of phrase types than the final experiment, with more "Imperative" feedback.

As in our main experiment, teachers often combined multiple forms of feedback. To separate them, we first split each message on punctuation (!.,;). We then labeled 685 interactions as one of five reference types which correspond to a particular form of feedback: "feature" for "Descriptive," "trajectory" for "Evaluative," "action-spatial" and "action-behavioral" for "Imperative," or "Other" if the utterance was unrelated to the task. We used scikit-learn to train a logistic regression on TF-IDF unigrams and bigrams, achieving a weighted-F1 of .86. For the purposes of our main experiment, we treated classifications of "object_behavior" as "Evaluative" feedback. Figure 8 shows the confusion matrix, and Table 2 shows full model training metrics. Code and data can be found in the **pilot_data_classifier_training** Jupyter notebook.

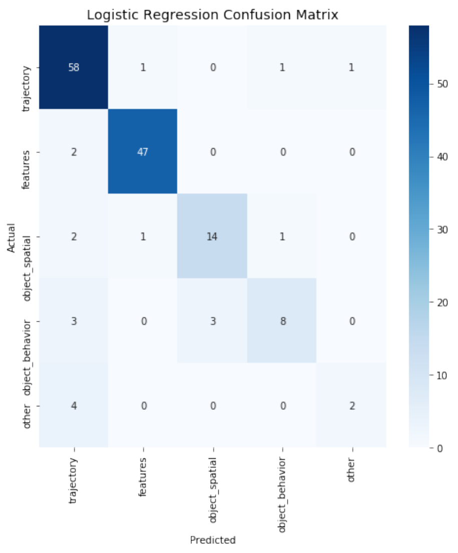


Figure 8: Test-set confusion matrix for classifier training. For results in the paper, the class "object_behavior" class was treated as "Evaluative" feedback.

	Precision	Recall	F1-Score	Support
Evaluative ("trajectory")	0.84	0.95	0.89	61
Descriptive ("features")	0.96	0.96	0.96	49
Imperative ("object_spatial")	0.82	0.78	0.80	18
Evaluative ("object_behavior")	0.80	0.57	0.67	14
Other	0.67	0.33	0.44	6
Accuracy			0.87	148
Macro Average	0.82	0.72	0.75	148
Weighted Average	0.87	0.87	0.86	148

Table 2: Full metrics for phrase-type classification used in the final paper.

2.5 Descriptive Feedback: Feature Lexicon

To map descriptive feedback to features, we used a small lexicon of features (see main text, Section 5.1). We enumerated all observed feature synonyms used in our corpus and regular expressions to identify these in the teachers' utterances. The full listing of synonyms is below:

- **Square**: square, box
- **Circle**: circle, dot
- **Triangle**: triangle, diamond
- **Blue**: blue, cyan, turquoise, aqua, teal
- **Pink**: pink, purple, magenta, violet
- **Yellow**: yellow, brown, gold, amber

2.6 Imperative Feedback: Action Designations

Teachers in our experiment rarely used "Imperative" language; when they did, it followed a fairly regular form. Almost universally, teachers used a variant of "top", "upper", "bottom", or "lower", plus "left" or "right" to designate one of the corners (e.g., Table 4; but also see Table 6, who used less consistent language). We thus matched on strings of the form:

$$(top|bottom|upper|lower)\ (right|left)$$

e.g., "top left", "lower right", and so on.

3 Inference Network Details

Code: is available in the **model_training** Jupyter notebook.

Data: is available in the *notebooks/data* subdirectory. Raw input data comes from the *human_trial_data.json* file. Because data augmentation takes 30 minutes to run on a Macbook Pro, we have also provided snapshots of per-fold processed training examples in the *notebooks/data/-model_training_10fold* subdirectory.

Models: pre-trained models are available in the *notebooks/data/model_training_10fold* subdirectory, in *.pkt* files according to the split they were trained on.

3.1 Data Filtering and Augmentation

We downsampled positive games to reduce skew in player scores. The majority of games played resulted in a positive score, so models trained on the whole dataset learned to predict the collected features as positive. To break this association, we first took all negative-scoring games in the dataset, then sampled an equal number of positive games. Finally, we took all zero-scoring games. To avoid learning a correlation between feature counts and positive scores, we downsampled positive games. We first took all negative-scoring rounds (156), sampled the same number of positive rounds, then took all zero-scoring rounds.

We used data augmentation to (1) prevent train-test split leakage of reward functions and (2) increase our training data size. We did this by retroactively switching the reward function used in the game, but preserving the relationship between the language, behavior, and underlying rewards. The augmentation procedure follows three steps: (1) choose a new reward function, (2) exchange feature-related chat tokens to reflect the new reward permutation, and (3) exchange the feature-counts. Concretely, we use the synonyms in Section 2.5 to identify direct feature references, then sample a synonym from the exchanged feature. For example, the utterance "Blue circles>blue squares>blue triangles" could have arisen under the reward function shown in Figure 3. If we chose a new reward function that swapped the valuation of circles and squares while maintaining other features, we would swap chat tokens accordingly and generate a phrase such as "Blue boxes>blue circles>blue triangles." We performed the same procedure on feature counts, thus maintaining the relationships between the language and underlying rewards. This increased our by a factor of approximately 29.

Executing the **model_training** Jupyter notebook will cause the original the training / validate / test splits to be loaded and used (contained in the *splits_*.pkl* files).

3.2 Training

We trained via stochastic gradient descent with a learning rate of .005. We stopped when we saw two consecutive steps with increased validation set error. We trained the entire model, including the word embeddings, end-to-end. Computational requirements were minimal; all models were trained on a Macbook Pro in under an hour.

3.3 Hyperparameters

Because our model was a proof-of-concept on a relatively small dataset, we did not perform extensive hyperparameter search. We describe the general set of tested hyperparameters below.

- **Network size.** We found that smaller architectures (32- and 64-wide) performed poorly. In particular, they were able to learn direct feature-associations but not praise.
- **Loss functions.** We experimented with L2 loss, L1 loss, and Huber loss. We did not see a significant effect on qualitative or quantitative network performance.
- **Regularization.** We experimented with both dropout and weight decay. Dropout caused significant instability in training. We tested a range of weight decay (0.005, 0.001, 0.0005, 0.0001). We found that minimal weight decay improved training stability; any more resulted in underfitting.

The full model definition is below:

```python
class TrajectoryFeedbackRewardPredictor(nn.Module):

    def __init__(self, vocab_size,
                 embedding_dim,
                 object_feature_dim,
                 reward_dim):

        super(TrajectoryOnlyRewardPredictor, self).__init__()

        self.embeddings = nn.EmbeddingBag(vocab_size, embedding_dim)

        full_input_width = embedding_dim + object_feature_dim

        self.linear1 = nn.Linear(full_input_width, 128)
        self.linear2 = nn.Linear(128, reward_dim)

    def forward(self, word_inputs, feature_inputs):

        embeds = self.embeddings(word_inputs)
        concated = torch.cat((embeds, feature_inputs.float()), 1)

        out = F.relu(self.linear1(concated))
        out = self.linear2(out).double()

        return out
```

Listing 1: Model Definition

4 Model Evaluation

4.1 Statistical Testing

Reproducibility: R code is in the "statistics.Rmd" notebook. The formatted data input, which comes from both of our experiments, is in "regression_input.csv."

We used Helmert encoding to perform the model comparison, with the first level comparing the Inference Network performance against the mean of the Literal and Pragmatic models. We used the second level to assess the contribution of the Pragmatic augmentations, comparing the Literal against the Pragmatic models. Both effects were significant, with $p < 1e-4$. See test output below.

Coefficients:

	Estimate	Std. Error	t value	Pr(>ltl)
(Intercept)	-3.5488	1.5915	-2.230	0.0258 *
Level Number	6.9173	0.2525	27.391	<2e-16 ***
Inference Net vs mean(Literal, Pragmatic)	3.8723	0.8967	4.318	1.61e-05 ***
Literal vs Pragmatic	-2.4436	0.5088	-4.803	1.61e-06 ***
Interactive vs Offline	3.1309	0.7769	4.030	5.67e-05 ***

Table 3: Statistical testing results validating model performance.

4.2 Human-Agent Experiment Analysis

Qualitatively, the human-agent experiment produced similar patterns of feedback, with more "Descriptive" overall. 50% of chats included a "Descriptive" component, 49% "Evaluative", and 6% "Imperative." There was a similar trend from "Descriptive" towards "Evaluative"; the difference in overall composition can likely be exlained by agents' overall lower scores, particularly in the first few rounds (Fig. 9).

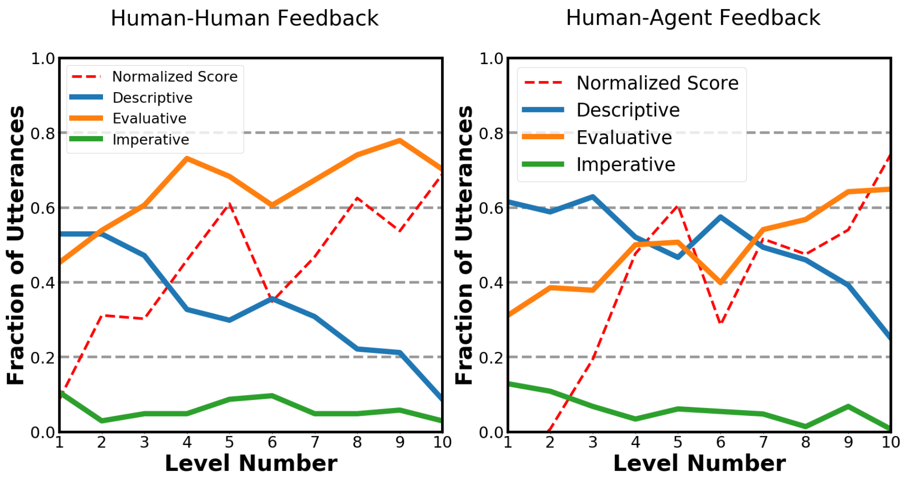


Figure 9: Feedback types and player scores for both experiments. The left image re-capitulates Fig 2 from the main paper; the right is the equivalent for the human-agent experiment.

4.3 Neural Net Performance

We analyze poor neural net performance on "Offline" pairs to understand its weaknesses. Observationally, the model struggled in three cases: when the teacher used *imperative* feedback or *negative* descriptive feedback (see Table 6), or when the teacher was nonsensical (see Table 7).

Imperative Feedback. Due to the relative sparsity of imperative feedback in the experiment (about 6% of utterances), however, we did not try to train the model to recognize it. As a result, the model performed poorly on teachers who used extensive imperative feedback (e.g. Table 6). Conceptually, it would be relatively straightforward to provide the model with the necessary features to handle this scenario. The model already sees feature counts from the trajectory, $\mu_\phi(\tau)$, and learns to associate praise or criticism (“good job”, “that was bad”) with those features. We could additionally include a vector describing possible actions (i.e. objects on the level) and a feature on each indicating their cluster assignment (e.g. objects in the top left cluster would have a “cluster ID” attribute == 1). We could then expect the model to learn to map tokens “top”, “upper”, etc to the appropriate objects.

Negative Descriptive Feedback. When teachers offered descriptive feedback, they tended to express *positive* sentiment about desirable features (e.g. Tables 4 and 5). As a result, our model generally learned to associate direct feature references with positive sentiment on those features (see the main paper, Figure 4). Because we used a simple mean bag-of-words representation, the model struggled to capture modifiers. As a result, when teachers used *negative* descriptive feedback (e.g. “*avoid* blue”, see Table 6) the model drew exactly the wrong conclusion: that blue was valuable. This issue could easily be alleviated by any kind of structured language model.

Nonsensical Teachers. We speculate that the model performed particularly poorly on nonsensical teachers (e.g. Table 7; note that the learned model underperformed even the custom models) because absent evidence, it tended to predict an average of the reward functions it had seen in training. These were (by design) *not* the reward functions in the evaluation data, and so the model, on average, made poor predictions. In contrast, the engineered models made mostly *neutral* predictions and so tended to score closer to zero, rather than extremely negative.

5 Interaction Examples

We provide player scores and raw text from several informative dyads below. Empty messages (sent with no text) are denoted <blank>, while multi-part utterances (i.e. sent with multiple carriage returns) are denoted with |. Scores for each model are reported in the caption.

Player Score (%)	Teacher Utterance
50	yellow squares and yellow circles. avoid everything else. yellow square are worth the most
25	go for the yellow triangles and squares. they are worth the most
100	nice job, perfect
100	that is absolutely spot on!
100	perfect again. great job
40	almost!
100	perfect
41	top right was the best choice. its okay!
100	perfect again!
100	great job. perfect choices again!

Table 4: **Successful teaching**. Final scores: Human: 150, Literal: 122, Pragmatic: 122, Inference Network: 135. A prototypical successful dyad. The teacher begins with descriptive feedback pointing to good features (levels 1-2), then switches to mostly evaluative praise (levels 3-10). When the learner scores poorly (level 8), they offer imperative feedback proscribing an alternative action.

Player Score (%)	Teacher Utterance
-31	only go for yellow shapes |
100	good job |
35	yellow circles are worth more than yellow triangles |
100	good job |
72	yellow circles are worth the most, then yellow squares, and yellow triangles are worth the least |
-60	go for yellow shapes |
100	good job |
100	good job |
100	good job |
100	good job |

Table 5: **Different feedback modes**. Final scores: Human: 123, Literal: 96, Pragmatic: 114, Inference Network: 84. Example of a teacher switching between different feedback modes depending on learner behavior. The teacher uses simple praise when the learner performs well, and descriptive feedback when the learner performs poorly.

Player Score (%)	Teacher Utterance
37	fo up-right avoid cyan |
100	go down-right | everything is safe
35	down-right, avoid blue
45	yellow circles are the highest value | | Go wherever there are the most yellow circles and avoid cyan/blue
72	The yellow triangles are worth about half as much as the yellow circles. So if you see two triangles, that's the same as one circle.
26	<blank>
100	The magenta/pink ones don't do any harm, but they don't really give you anything either. |
100	Good goin'!
96	Yeah, upper-left would be the real winner here, right. Two yellow circles, three yellow squares. And the cyan/blue is death. |
100	<blank>

Table 6: **Poor Inference Network performance**. Final scores: Human: 133, Literal: 125, Pragmatic: 78, Inference Network: -61. A particularly informative example of a poor performance from the learned model: this teacher used *both* imperative language ("go down-right") and negative descriptive feedback ("avoid blue"). The learned model could not use the imperative feedback, and negative descriptive feedback was extremely rare in the data. As a result, the network inferred *positive* rewards for blue.

Player Score (%)	Teacher Utterance
0	seemed like i had no control
-56	i do not understand any longer
35	oh well! |
0	...
80	ok
26	sigh!
-121	dont you talk? |
100	tired
44	I AM TIRED!
100	GO ON |

Table 7: **Useless teacher**. Final scores: Human: 48, Literal: 26, Pragmatic: -12, Inference Network: -60. The teacher clearly did not understand the task; the human learner was evidently able to recover some of the reward function from their per-level scores.

Player Score (%)	Teacher Utterance
-12	that time turquoise was worth most
68	nice! turquoise looking good |
-11	avoid violet and stick with the light blue. Also light blue squares are worth more than circles |
-75	okaaay you do you
20	only light blue if you can | |
26	<blank>
-121	go for violet and yellow now |
83	cool
40	<blank>
75	look at that fresh 9 cents whoa |

Table 8: **Useless (human) learner**. Final scores: Human: 17, Literal: 85, Pragmatic: 73, Inference Network: -37. Example of a learner who didn't listen to the teacher.

Player Score (%)	Teacher Utterance
31	try going towards the blue squares |
100	nice | |
35	The blue squares are worth 8-9 points ($$) as opposed to the triangles and circles (3-5) |
100	EXCELLENT! | |
72	Avoid the yellow squares and circles. Go for the blue squares (most points), blue circles (second most points), and lastly triangles (least points) |
26	Try to always head in the direction of the blue square if possible. They have the biggest bonsues($$). Then circles. Triangles have only 1-3 points
64	MUCH BETTER!
100	Well done! You're doing great! |
100	YES!!!!!!! |
55	Great job! | | I'm proud! |

Table 9: **Quantitative values**. Final scores: Human: 132, Literal: 99, Pragmatic: 130, Inference Network: 141. Example of a (rare) teacher who provided actual point values instead of general preferences or relative values.

Player Score (%)	Teacher Utterance
0	get the purple squares
0	get the purple squares and triangles
100	nice
0	always get the purple squares and triangles l
-24	get the purple squares and purple triangles
0	get the purple squares and triangles if you want to make money
0	too slow
0	oh well lol
40	nice job
0	have a nice day

Table 10: **Sentiment pragmatics**. Final scores: Human: 22, Literal: 56, Pragmatic: 120, Inference Network: 104. The teacher referenced features repeatedly without an explicit sentiment, so the "Literal" model inferred a latent reward of 0 on pink / purple.