# INTERACTIVE CLASSIFICATION BY ASKING INFORMATIVE QUESTIONS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We propose an interactive classification approach for natural language queries. Instead of classifying given the natural language query only, we ask the user for additional information using a sequence of binary and multiple-choice questions. At each turn, we use a policy controller to decide if to present a question or provide the user the final answer, and select the best question to ask by maximizing the system information gain. Our formulation enables bootstrapping the system without any interaction data, instead relying on non-interactive crowdsourcing annotation tasks. Our evaluation shows the interaction helps the system increase its accuracy and handle ambiguous queries, while our approach effectively balances the number of questions and the final accuracy.[1]

## 1 INTRODUCTION

Responding to natural language queries through simple classification has been studied extensively, including for question answering (Rajpurkar et al., 2016; Chen et al., 2017; Yang et al.) and information retrieval (Balcan et al., 2008; Ailon & Mohri, 2007). The common approach is to return a response given the input query only. This strategy misses the opportunity to interact with the user to improve the system behavior. For example, a user may provide an underspecified request due to partial understanding of the domain or the system, or the system may fail to fully interpret the nuances of the natural language input. In both cases, given the query alone, the system will likely return a low quality response, which could have been improved by obtaining additional information.

In this paper, we propose to recover from such failures through a simple but effective interaction, where the system asks the user for additional information using a sequence of binary and multiple-choice questions. We design a simple method that combines the benefits of such interaction, but avoids much of the complexity and challenges involved in supporting unrestricted natural language interaction. Figure 1 illustrates the type of interaction our method enables in our two evaluation domains. We study an interaction class that supports obtaining the required information from the user, but avoids much of the challenges of full fledged dialogue. Such an interaction begins with a natural language query from a user. The system then decides if to return the classification output label, or pose a question to obtain more information. Given the user response, the system decides if to ask another question, or conclude the interaction and return a response to the user.

We emphasize two aspects in our approach: interaction efficiency and simple system building. To make the interaction efficient, we maintain a posterior distribution over classification labels that we update during the interaction. We select the next question to ask at each turn by computing the information gain about this distribution from observing the answer to each available question, and train a policy controller to decide if to ask a question or return a prediction. In building our system, we emphasize avoiding collecting expensive interaction data, which is often done by tuning an automated system (Wu et al., 2018; Hu et al., 2018; Lee et al., 2018; Rao & Daumé III, 2018) or Wizard of Oz studies (Kelley, 1984; Wen et al., 2016). We propose a simple approach to crowdsource initial natural language queries and question-answer pairs for each classification label. This data enables training the system without any human interaction data.

We evaluate our method on two public datasets, FAQ document suggestion (Shah et al., 2018) and bird species identification using the text data of the CUB-200 dataset (Wah et al., 2011). Our exper-

[1] Our code, data, and experiment setup will be made publicly available.

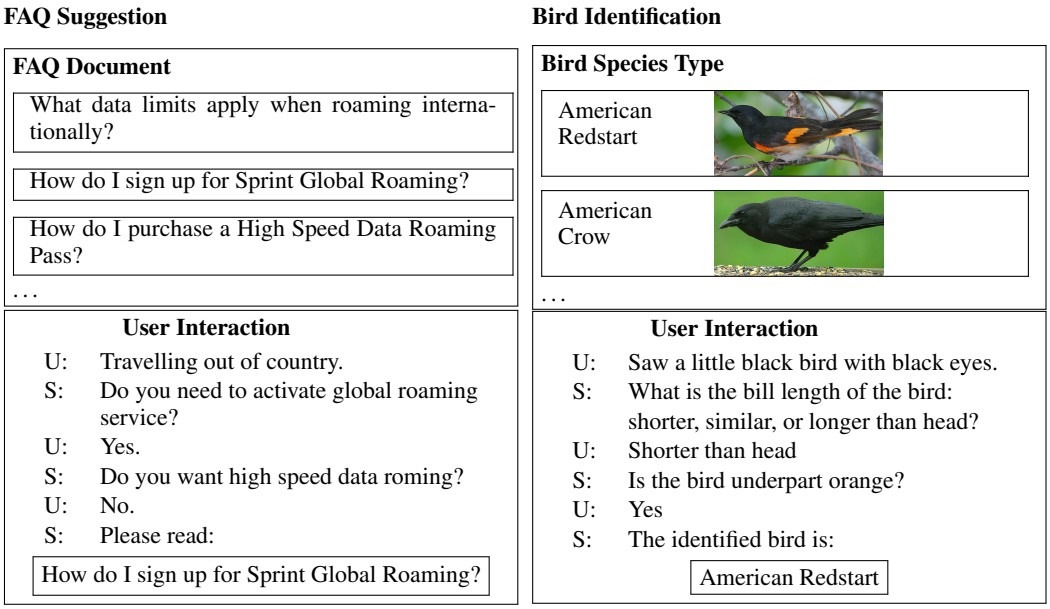


Figure 1: Example interactions in the FAQ (left) and Birds (right) domains. The top boxes show example classification output labels: FAQ documents or bird species. The lower boxes show a user (U) interacting with the system (S). The user starts with an initial natural language query. At each step, the system asks a clarification question. The interaction ends with the system returning an output labels. The images are for illustration only. Our model does not consider images.

iments shows that interaction increases the classification accuracy. With one clarification question, our system gains a relative accuracy boost of $40\%$ and $65\%$ for FAQ suggestion and bird identification compared to no-interaction baselines on simulator evaluation. Given at most five turns of interaction, our approach improves accuracy by over $100\%$ on both tasks for both simulator and human evaluation.

## 2 Technical Overview

**Task** Our goal is to classify a natural language user query to one of a set of labels through an interaction. Let $\mathcal{Y}$ be a set of $N$ labels $\{y^{(1)}, \ldots, y^{(N)}\}$. In the FAQ domain, each $y$ is an FAQ document, and in the Birds domain, each $y$ is a bird specie. To simplify our notation, we consider the label as the text representation of the underlying object: the FAQ text or the bird name. A classification interaction $x$ is a tuple $(x_0, \langle (q_1, r_1), \ldots, (q_T, r_T) \rangle, y)$, where $x_0$ is the initial natural language query, $\langle (q_1, r_1), \ldots, (q_T, r_T) \rangle$ is a sequence of questions $q_i$ and user responses $r_i$, and $y$ is the final classification output returned to the user. There are two types of questions: binary and multiple choice. The predfined set of possible answers for a question $q$ is $\mathcal{R}(q)$. The possible answers for binary questions are "yes" and "no", and for multiple choice questions a predefined set of question-specific values. We denote an interaction up to time $t$ as $x_t = (x_0, \langle (q_1, r_1), \ldots, (q_t, r_t) \rangle)$. Figure 1 shows interactions in our two evaluation domains.

**Model** The aim of the interaction is to improve the system accuracy over using only the initial query for classification. We model the probability of a label $y \in \mathcal{Y}$ at time $t$ in the interaction using the parameterized distribution $p(y \,|\, x_{t-1})$, where $x_{t-1}$ is the interaction until and including step $t-1$. We use information gain over this probability distribution to select the next question to ask $q_t$, and a policy controller to decide between returning the current most likely label $y^* = \arg\max_y p(y \,|\, x_{t-1})$ or present the next question to the user. If the policy decides to present the question $q_t$, we use the user answer $r_t$ to compute $p(y \,|\, x_t)$. Section 4 describes our model.

**Learning** We assume access to a dataset $\{(y^{(i)}, X^{(i)}, \{(q_j, r_j)\}_{j=1}^{M^{(i)}})\}_{i=1}^{N}$, where each label $y^{(i)}$ is annotated with a set of initial queries $X^{(i)}$ and a set of $M^{(i)}$ question-answer pairs $\{(q_j, r_j)\}_{j=1}^{M^{(i)}}$. The pairs in the set $\{(q_j, r_j)\}_{j=1}^{M^{(i)}}$ are independent from each other, and do not form an interaction. We crowdsource this data by presenting workers with a target label $y$ and asking for initial queries and text tags describing $y$. We use the initial queries as is, and deterministically process the tags to

create question-answer pairs. For a question $q$, we denote its source tag as $\bar{q}$. We describe the data collection in Section 5. We use this data to train our model (Section 4.3), create a user simulator (Section 4.4), and to train the policy controller using policy gradient (Section 4.5). The policy is trained to minimize the number of interaction turns while achieving high classification accuracy.

**Evaluation** We evaluate classification accuracy, and study the trade-off between accuracy and the number of the turns the system takes. We use both human evaluation and our simulator. For human evaluation, we additionally collect qualitative ratings of the quality of the interaction.

## 3 Related Work

**Interactive Learning** A number of recent studies have leveraged human feedback to train machine learning systems, including for dialogue learning (Li et al., 2016), semantic parsing (Artzi & Zettlemoyer, 2011; Wang et al., 2016; Iyer et al., 2017), text classification (Hancock et al., 2018), and SQL generation (Gur et al., 2018). We study the benefit of interaction for classification. In contrast to this line of work, we train our system without access to full interactions. This makes our training data less costly to obtain, as it does not require building a working system (Wu et al., 2018; Hu et al., 2018; Lee et al., 2018; Rao & Daumé III, 2018) or conducting Wizard of Oz experiments (Kelley, 1984; Wen et al., 2016). Rich full interaction annotation was also used in other interactive systems, including for visual question answering (De Vries et al., 2016; Lee et al., 2018; Chattopadhyay et al., 2017; Das et al., 2017) and multi-turn text-based question answering (Rao & Daumé III, 2018; Reddy et al., 2019; Choi et al., 2018). Collecting this data is usually done by building an interactive multi-worker system on a crowdsourcing platform, a similar approach to Wizard of Oz studies.

Recently, Chung et al. (2018) proposed a system for interactive spoken content retrieval, focusing on learning a user simulator given access to the target documents as training signal. In contrast, our aim is to learn a complete system for interacting with real users. The simulator building methods they present are orthogonal to ours, and provide a more costly way to build a stronger simulator for our learning approach. Our classification problem can be viewed as an instance of the popular 20-question game (20Q), which has been studied recently using a celebrity knowledgebase (Chen et al., 2018; Hu et al., 2018). Our setup differs in that our interactions begin with an initial natural language query, and the interaction goal is to refine the system response to it (i.e., return the right classification label). In addition, we do not assume access to a structured knowledgebase, instead relying only a collection of text documents.

**Matching-based Classification** Classification methods by matching the input text and the label text have been extensively studied and used in various natural language processing applications, such as few-shot classification (Yu et al., 2018), forum-based question answering (dos Santos et al., 2015; Lei et al., 2015; Rao & Daumé III, 2018), and dialogue response selection (Lowe et al., 2015; Zhou et al., 2016; Wu et al., 2017; Zhou et al., 2018). While previous works mostly focus on improving the model components for better non-interactive text matching (Wang & Manning, 2010; Heilman & Smith, 2010; Dilek et al., 2018; Rajpurkar et al., 2016), we propose a different modeling and learning method to proactively inquire information from the users, which can be combined with previous methods to improve the overall performance.

## 4 Method

We maintain a probability distribution $p(y \mid x_t)$ over the set of labels $\mathcal{Y}$. At each interaction step, we first update this belief, select a question to ask using information gain, and decide if to ask the question or return the classification output using a policy controller.

### 4.1 Belief Probability Decomposition

We decompose the distribution $p(y \mid x_t)$ using Bayes rule and assuming independence between turns in the interaction $x_t = (x_0, \langle q_1, r_1 \rangle, \ldots, \langle q_t, r_t \rangle)$:

$$\begin{aligned} p(y \mid x_t) &= p(y \mid x_{t-1}, q_t, r_t) \\ &\propto p(r_t, q_t, y \mid x_{t-1}) \\ &= p(r_t \mid q_t, y, x_{t-1}) \cdot p(q_t \mid y, x_{t-1}) \cdot p(y \mid x_{t-1}) \ . \end{aligned} \tag{1}$$

We use a deterministic process to select the next question $q_t^*$ given the history of the interaction $x_{t-1}$. In Section 4.2, we describe how we implement this process using information gain. Because

of the independence assumption between turns, we can write $p(r_t \,|\, q_t, y, x_{t-1}) = p(r_t \,|\, q_t, y)$. This allows to create an incremental update rule:

$$p(y \,|\, x_t) \;\propto p(r_t \,|\, q_t, y) \cdot p(y \,|\, x_{t-1}) \cdot \mathbb{1}[q_t = q_t^*] = p(y \,|\, x_0) \prod_{j=1}^{t} p(r_j \,|\, q_j, y) \;\; , \tag{2}$$

where $\mathbb{1}[\cdot]$ is an indicator function and $\mathrm{p}(y \,|\, x_0)$ is the label distribution given the initial query only. This factorization allows us to leverage separate annotations to train $\mathrm{p}(y \,|\, x_0)$ and $\mathrm{p}(r \,|\, q, y)$ directly, which alleviates the need for collecting costly user interactions.

### 4.2 Question Selection using Information Gain

The system selects the question $q_t^*$ to ask at turn $t$ by maximizing its information gain. To efficiently compute the information gain, we decompose it to two quantities that we incrementally update during the interaction: $p(r_t \,|\, x_{t-1}, q_t)$ and $p(y \,|\, x_{t-1}, q_t, r_t)$. Given $x_{t-1}$, we compute the information gain for the target label random variable $Y$ by observing the answer to question $q_t$:

$$IG(Y \,;\, q_t \,|\, x_{t-1}) = H(Y \,|\, x_{t-1}) - H(Y \,|\, x_{t-1}, q_t) \;\; ,$$

where $H(\cdot \,|\, \cdot)$ denotes the conditional entropy. Because the first entropy term $H(Y \,|\, x_{t-1})$ is a constant regardless of the choice of $q_t$, the selection of $q_t^*$ is equivalent to:

$$q_t^* = \arg\min_{q_t} H(Y \,|\, x_{t-1}, q_t) = \arg\min_{q_t} \sum_{r_t \in \mathcal{R}(q_t)} p(r_t \,|\, x_{t-1}, q_t) \cdot H(Y \,|\, x_{t-1}, q_t, r_t) \;\; ,$$

where $\mathcal{R}(q_t)$ is the set of answers for question $q_t$. Because of the independence between turns and the deterministic selection of $q_t$ given $x_{t-1}$:

$$p(r_t \,|\, x_{t-1}, q_t) = \sum_{y \in \mathcal{Y}} p(r_t, y \,|\, x_{t-1}, q_t) = \sum_{y \in \mathcal{Y}} p(r_t \,|\, q_t, y) \cdot p(y \,|\, x_{t-1}) \;\; ,$$

and

$$H(Y \,|\, x_{t-1}, q_t, r_t) = \sum_{y \in \mathcal{Y}} p(y \,|\, x_{t-1}, q_t, r_t) \cdot \log p(y \,|\, x_{t-1}, q_t, r_t) \;\; .$$

Both $p(r_t \,|\, x_{t-1}, q_t)$ and $p(y \,|\, x_{t-1}, q_t, r_t)$ can be efficiently updated as the interaction progresses using Equations 1 and 2.

### 4.3 Modeling the Distributions

We model $p(y \,|\, x_0)$ and $p(r \,|\, q, y)$ using a simple recurrent neural network (SRU; Lei et al., 2018) encoder $\mathbf{enc}(\cdot)$ with parameters $\psi$. We use the same encoder to encode all texts. Both probability distributions are computed using the scoring function $S(u, v) = \mathbf{enc}(u)^\top \mathbf{enc}(v)$, where $u$ and $v$ are two pieces of text that are scored.

The probability of predicting the label $y$ given an initial query $x_0$ is:

$$p(y \,|\, x_0) = \frac{\exp(S(y, x_0))}{\sum_{y' \in \mathcal{Y}} \exp(S(y', x_0))} \;\; .$$

The probability of an answer $r$ given a question $q$ and label $y$ is a linear combination of the observed empirical distribution $\hat{p}(r \,|\, q, y)$ and a parameterized estimate $\tilde{p}(r \,|\, q, y)$:

$$p(r \,|\, q, y) = \lambda \cdot \hat{p}(r \,|\, q, y) + (1 - \lambda) \cdot \tilde{p}(r \,|\, q, y) \;\; ,$$

where $\lambda \in [0, 1]$ is a hyper-parameter. We use the question-answer annotations for each label $y$ to calculate the empirical distributions to estimate $\hat{p}(r \,|\, q, y)$. For example, in the FAQ task, we collect multiple user responses for each label and question pair, and average across annotator answers to estimate $\hat{p}$ (Section 5). The second term $\tilde{p}(r \,|\, q, y)$ is:

$$\tilde{p}(r \,|\, q, y) = \frac{\exp(w \cdot S(\bar{q} \# r, y) + b)}{\sum_{r' \in \mathcal{R}(q)} \exp(w \cdot S(\bar{q} \# r', y) + b)} \;\; ,$$

where $w, b \in \mathbb{R}$ are scalar parameters and $\bar{q}\#r$ is a concatenation of the tag of the question $q$ and the $r$. Because we do not collect sufficient annotations to cover every label-question pair, the combination of the two terms provides a smoothing of the observed counts that leverages the learned encoder through the function $S(\cdot)$.

The parameters $w$ and $b$ are randomly initialized and estimated using reinforcement learning while training the policy controller (Section 4.5). We estimate the $\mathbf{enc}(\cdot)$ parameters $\psi$ by pre-training using a dataset, $\{(y^{(i)}, X^{(i)}, \{(q_j, r_j)\}_{j=1}^{M^{(i)}})\}_{i=1}^{N}$, where each target label $y^{(i)}$ is paired with a set of annotated initial queries $X^{(i)}$ and question-answer pairs $\{(q_j, r_j)\}_{j=1}^{M^{(i)}}$. We use this data to create a set of text pairs $(u, v)$ to train the scoring function $S(\cdot)$. For each label $y^{(i)}$, we create pairs $(x_0, y^{(i)})$ with all its initial queries $x_0 \in X^{(i)}$. We also create $(\bar{q}\#r, y^{(i)})$ for each question-answer pair $(q, r)$ associated with the label $y^{(i)}$. We perform gradient descent to minimize the cross-entropy loss:

$$\mathcal{L}(\psi) = -S(u, v) + \log \sum_{v'} \exp(S(u, v')) \quad .$$

The second term requires summarition over all $v$-s, which are all the labels in $\mathcal{Y}$. We approximate this sum using negative sampling that replaces the full set $\mathcal{Y}$ with a sampled subset in each training batch.

### 4.4 User Simulator

The user simulator provides initial queries to the system and emulates user responses to the system initiated clarification questions. The simulator is based on two distributions $p'(x_0 \,|\, y)$ and $p'(r \,|\, q, y)$, where $x_0$ is an initial query, $y$ is a target label, $q$ is a system question, and $r$ is a user response. We estimate the two distributions using smoothed empirical counts from held-out set $\{(y^{(i)}, X^{(i)}, \{(q_j, r_j)\}_{j=1}^{M^{(i)}})\}_{i=1}^{N'}$ of tuples of goal $y^{(i)}$, set of initial queries $X^{(i)}$, and question-answer pairs $(q_j, r_j)$. While this data is identical in structure to our training data, we keep it separate from the data used to estimate the scoring function $S(\cdot)$ (Section 4.3).

At the beginning of an interaction, the simulator selects a target label $y$, and samples from the associated query set a query $x_0$ to start the interaction. Given a system clarification question $q_t$ at turn $t$, the simulator responds with an answer $r_t \in \mathcal{R}(q_t)$ by sampling from a belief probability $p'(r_t \,|\, q_t, y)$. Sampling provides natural noise to the interaction, and our model has no knowledge of $\tilde{\mathrm{p}}$. The interaction ends when the system returns a target. This setup is flexible in that the user simulator can be easily replaced or extended by real human, and the system can be further trained with the human-in-the-loop setup.

### 4.5 Policy Controller

The policy controller decides at each turn $t$ to either select another question to query the user or to conclude the interaction. This provides a trade-off between exploration by asking questions and exploitation by returning the highest probability classification label. The policy controller $f(\cdot, \cdot; \theta)$ is a feed-forward network parameterized by $\theta$ that takes the top-$k$ probability values and current turn $t$ as input state. It generates two possible actions, ***STOP*** or ***ASK***. When the action is ***ASK***, a question is selected to maximize the information gain, and when the action is ***STOP***, the highest probability label $y$ is returned using $\arg\max_{y \in \mathcal{Y}} \mathrm{p}(y \,|\, x_t)$.

We tune the policy controller using the user simulator (Section 4.4). Algorithm 1 describes the training process. During learning, we use a reward function that provides a positive reward for predicting the correct target at the end of the interaction, a negative reward for predicting the wrong target, and a small negative reward for every question asked. We learn the policy controller $f(\cdot, \cdot; \theta)$, and fine-tune $p(r \,|\, q, y; \psi)$ by back-propagating through the policy gradient. We keep the $\mathbf{enc}(\cdot)$ parameters fixed during this process.

**Algorithm 1:** Full model training

Initialize text encoder, model for $p(y \mid x_0)$, and user simulator SIM
**for** *episode = 1 .. M* **do**
  Sample $(x_0, \hat{y})$ from dataset
  **for** *t = 1 .. T* **do**
    Compute $p(y \mid x_{t-1})$ (Equation 2)
    action $= f(p(y \mid x_{t-1}), t-1; \theta)$
    **if** action *is STOP* **then**
      break
    **else if** *action is ASK* **then**
      $q_t = \arg\max_{q_t \in \mathcal{Q}} \; IG(Y \,;\, q_t \mid x_{t-1})$
      $r_t = \text{SIM}(q_t, \hat{y})$
  **end**
  $y = \arg\max_y \; p(y \mid x_t)$
  reward $= \text{SIM}(y, \hat{y})$
  Update $w$, $b$, $\theta$ using policy gradient
**end**

## 5 Data Collection

We design a crowdsourcing process that does not require collecting interaction data, and use it to collect data for the FAQ domain using Amazon Mechanical Turk[2]. For the Birds domain, we re-purpose an existing dataset. We collect initial queries and tags for each FAQ document.

**Initial Query Collection** We ask workers to consider the scenario of searching for an FAQ supporting document using an interactive system. Given a target FAQ, we ask for an initial query they would provide such a system. The set of initial queries collected for each document $y^{(i)}$ is $X^{(i)}$. We encourage workers to provide incomplete information intentionally. This results in natural diverse utterances that enable learning a robust system. For example, given a target FAQ "How do I sign up for Sprint Global Roaming?", a worker may write "Travelling out of country". The workers do not engage in a multi-round interactive process. This allows for cheap and scalable collection.

**Tag Collection** We collect natural language tag annotations for the FAQ documents. First, we use domain experts to define the set of possible free-form tags. The tags are not restricted to a predefined ontology and can be a phrase or a single word. The tags describe the topic of the document. We heuristically remove duplicate tags to finalize the set. We use a small set of deterministic, heuristically designed templates to convert tags into questions. For example, tag "international roaming" will be converted into question "Is it about international roaming?". As part of the post-processing steps, an experts puts each tag into one of the binary or mutli-choice category, based on the nature of it template-generated question. We then use non-experts to associate relevant tags to the FAQ documents. For binary tags, we show the workers a list of ten tags for a given target as well as an "none of the above" option. Annotating all target-tag combinations is excessively expensive and most pairings are negative. We rank the tags based on the relevance against the target using $S(\cdot)$ and show only top-50 to the workers. For multi-choice tags, we show the workers a list of possible answers to a tag generated question for a given FAQ. The workers need to choose one answer that they think best applies. They also have the option of choosing "not applicable". We provide more data collection statistics in Appendix A.1.

## 6 Experimental Setup

**Task I: FAQ Suggestion** We use the FAQ dataset from Shah et al. (2018). The dataset contains $517$ troubleshooting documents crawled from Sprint technical website. In addition, we collect $3,831$ initial queries and $118,640$ tag annotations using the setup described in Section 5. We split the data into $310/103/104$ documents as training, development, and test sets. Only the queries and tag annotations of the $310$ training documents are used for pre-training and policy learning. We use

[2]https://www.mturk.com/

the queries and tag annotations of the development and test documents for evaluation only. The classification targets contain all 517 documents during evaluation[3].

**Task II: Bird Identification** Our second set of experiments use the Caltech-UCSD Birds (CUB-200) dataset Wah et al. (2011). The dataset contains $11,788$ bird images for 200 different bird species. Each bird image is annotated with a subset of 27 visual attributes and 312 attribute values pertaining to the color or shape of a particular part of the bird. We take attributes with value count less than 5 as categorical tag (8 categorical questions) and the rest as binary tag (279 binary questions). In addition, each image is annotated with 10 image captions describing the bird in the image Reed et al. (2016). Since each image often contains only only partial information about the bird species, the data is naturally noisy and provides challenging user scenarios. In our experiments, we use the image captions as user initial queries and bird class names as targets. We do not use the images of the dataset for modeling, and only as the user scenarios during human evaluation.

**Baselines** We compare our full model (OURS) against the following baseline methods to isolate and evaluate of our model components: $(a)$. NO INTERACT: The best classification target is predicted using only the initial query, according to belief distribution $\mathrm{p}(y \mid x_0)$. We consider two possible implementations. The first one is BM25, a common keyword based scoring model for retrieval methods Robertson & Zaragoza (2009). The second implementation is our neural model described in Section 4.3. $(b)$. RANDOM INTERACT: At each turn, a random question is chosen and presented to the user. After $T$ turns the best target is chosen according to the final belief $\mathrm{p}(y \mid x_T)$. $(c)$ STATIC INTERACT: Use the same maximum information criterion to pick questions but without conditioning on the initial query, similar to (Utgoff, 1989; Ling et al., 2004)

We also consider several variants of our full model. The first two variants replace the policy controller with two termination strategies using either a threshold on $\mathrm{p}(y \mid x_t)$ or interact only up to a predefined number of turns $T$. The third variant disables the parameterized estimator $\hat{\mathrm{p}}(r \mid q, y)$ by setting $\lambda$ as 1.

**Evaluation** Given the user simulator, we evaluate the classification performance of our model and all baselines using Accuracy@k, which is the percentage of time the correct target appears among the top-k predictions of the model. In addition, we conduct human evaluation by asking annotators to interact with our model or baseline methods through a Web based interactive interface. Each interaction session starts with presenting the annotator an user scenario (e.g a bird image or an issue with your phone). Once the system returns the final target, the annotator is asked to provide a few ratings of the interaction, such as *rationality – do you feel being understood by the system?*. We present more details of the human evaluation in Appendix A.4.

**Implementation Details** The policy controller receives three different rewards – a positive reward for correctly returning the correct target ($r_p = 20$), a negative reward for providing the wrong target ($r_n = -10$) and a small negative reward for each turn used ($r_a = -1, \ldots, -5$). We report the averaged results over 3 independent runs for each model and baseline. More details about the model implementation and training procedure can be found in Appendix A.2.

| | FAQ Suggestion | | Bird Identification | |
|---|---|---|---|---|
| | Acc@1 | Acc@3 | Acc@1 | Acc@3 |
| NO INTERACT (BM25) | 26% | 31% | N.A. | N.A. |
| NO INTERACT (neural) | 38% | 61% | 23% | 41% |
| RANDOM INTERACT | 39% | 62% | 25% | 44% |
| STATIC INTERACT | 46% | 66% | 29% | 50% |
| OURS | **79%** | **86%** | **49%** | **69%** |
| Threshold | 72% | 82% | 40% | 59% |
| Fixed Turn | 71% | 81% | 39% | 56% |
| with $\lambda = 1$ | 66% | 71% | 40% | 60% |

Table 1: Performance of our system against various baselines, which are evaluated using Accuracy@$\{1, 3\}$. For all interacting baselines, 5 clarification questions are used.

[3]The target classes from development and test sets are hidden to the model during training. This split is set up to ensure the model to generalize to newly added or unseen FAQs.

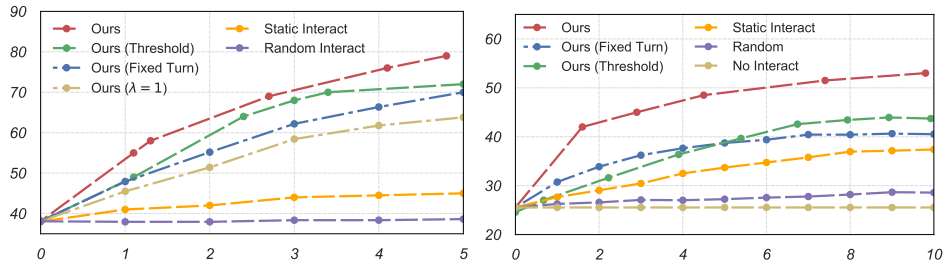


Figure 2: Accuracy@1 (y-axis) against turns of interactions (x-axis) for FAQ suggestion (left) and Bird identification (right) tasks

| | FAQ Suggestion | | | Bird Identification | | |
|---|---|---|---|---|---|---|
| | Count | Acc@1 | rationality | Count | Acc@1 | rationality |
| OURS | 60 | **59%** | **0.81** | 60 | **45%** | **0.95** |
| OURS (Fixed Turn) | 55 | 56% | 0.45 | 55 | 37% | 0.55 |
| STATIC INTERACT | 55 | 45% | 0.41 | 55 | 28% | 0.85 |

Table 2: Human evaluation results. Count is the total number of interaction examples. The system is evaluated with Accuracy@1 and the rationality score ranging from $-2$ (strongly disagree) to 2 (strongly agree).

## 7 RESULTS

**Simulator Evaluation** Table 1 shows the performance of our model against the baselines on both tasks while evaluating against user simulator. The NO INTERACT (neural) baseline achieves a Accuracy@1 of 36% and 23% on FAQ and Birds domains, respectively. The NO INTERACT (BM25) baseline performs worst. The RANDOM INTERACT baseline and the STATIC INTERACT baseline barely improves the performance from interactions, illustrating the challenge of building an effective interactive model. In contrast, our model and its variants obtain substantial gain in accuracy given a few number of interactions. Our full model achieves a Accuracy@1 of 78% and 49% using less than 5 turns, on FAQ and Birds respectively, outperforming the NO INTERACT (neural) baseline by an absolute number of 40% and 26%. The two baselines with alternative termination strategies underperform the full model, indicating the effectiveness of policy controller trained with reinforcement learning. The model variant with $\lambda = 1$, which has fewer probability components leveraging natural language than our full model, achieves much worse Accuracy@1. This result, together with the fact that our model outperforms the STATIC INTERACT baseline, confirms the importance of modeling natural language for efficient interaction.

Figure 2 shows how model accuracy with different values of the turn penalty in the reward. For the threshold model baseline, we show performance for different thresholds, and for the fixed-turn baseline we show performance for different number of turns. Interactions with our full model with either the policy controller or threshold strategy vary in length (i.e., number of turns) depending on the interaction progress. We report an averaged number of turns across multiple runs for these two models. With one clarification question, we achieve a relative accuracy boost of 40% and 65% for FAQ suggestion and bird identification over no-interaction baselines, indicating the value of human feedback in classification tasks.

**Human Evaluation** Table 2 shows the human evaluation results of our full model and two baselines on the FAQ and Birds tasks. Each of the model variants uses 3 interaction turns on average, and all three models improve the classification result after the interaction. Our full model obtains the best performance, achieving 64% and 64% for Accuracy@1 on FAQ suggestion and bird identification tasks. In addition, users rate our full as more rational. The human evaluation demonstrates that our model handles interaction with real users effectively and that the interaction improves classification accuracy, despite training with only non-interactive data. Appendix A.4 includes additional details of the human evaluation and example interactions.

## 8 CONCLUSION

We propose an approach for interactive classification, where users can provide under-specified natural language queries and the system can inquire missing information through a sequence of simple binary questions. Our method uses information theory to select the best question at every turn, and a lightweight policy to efficiently control the interaction. We show how we can bootstrap the system without any interaction data. We demonstrate the effectiveness of our approach on two tasks with different characteristics. Our results show that our approach outperforms multiple baselines by large margin. In addition, we provide a new annotated dataset for future work on bootstrapping interactive classification systems.

## REFERENCES


Nir Ailon and Mehryar Mohri. An efficient reduction of ranking to classification. *arXiv preprint arXiv:0710.2889*, 2007.

Yoav Artzi and Luke Zettlemoyer. Bootstrapping semantic parsers from conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 421–432, 2011. URL http://www.aclweb.org/anthology/D11-1039.

Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin. Robust reductions from ranking to classification. *Machine Learning*, 72 (1):139–153, Aug 2008. ISSN 1573-0565. doi: 10.1007/s10994-008-5058-6. URL https://doi.org/10.1007/s10994-008-5058-6.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. arXiv:1607.04606, 2016. URL https://arxiv.org/abs/1607.04606.

Prithvijit Chattopadhyay, Deshraj Yadav, Viraj Prabhu, Arjun Chandrasekaran, Abhishek Das, Stefan Lee, Dhruv Batra, and Devi Parikh. Evaluating Visual Conversational Agents via Cooperative Human-AI Games. 2017. URL https://arxiv.org/pdf/1708.05122.pdf.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051, 2017. URL http://arxiv.org/abs/1704.00051.

Yihong Chen, Bei Chen, Xuguang Duan, Jian-Guang Lou, Yue Wang, Wenwu Zhu, and Yong Cao. Learning-to-ask: Knowledge acquisition via 20 questions. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1216–1225. ACM, 2018.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. Quac : Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. URL http://aclweb.org/anthology/D18-1241.

Pei-Hung Chung, Kuan Tung, Ching-Lun Tai, and Hung-Yi Lee. Joint learning of interactive spoken content retrieval and trainable user simulator. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association*, 2018. URL https://arxiv.org/pdf/1804.00318.pdf.

Abhishek Das, Satwik Kottur, Jos M F Moura, Stefan Lee, and Dhruv Batra. Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning. 2017. URL http://openaccess.thecvf.com/content_ICCV_2017/papers/Das_Learning_Cooperative_Visual_ICCV_2017_paper.pdf.

Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin Deepmind, Hugo Larochelle Twitter, and Aaron Courville. GuessWhat?! Visual object discovery through multi-modal dialogue. 2016. URL http://openaccess.thecvf.com/content_cvpr_2017/papers/de_Vries_GuessWhat_Visual_Object_CVPR_2017_paper.pdf.

Kezban Dilek, Ye Zhang, Ismail Sengor AltingovdeMd, Mustafizur RahmanPinar Karagoz, Alex Braylan, Brandon Dang, Heng-Lu Chang, Henna Kim, Quinten McNamara, Aaron Angert, Edward Banner, Vivek Khetan, Tyler McDonnell, An Thanh Nguyen, Dan Xu, Byron C. Wallace, Maarten de Rijke, and Matthew Lease. Neural information retrieval: at the end of the early years. *Information Retrieval Journal*, 2018.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, 2015.

Izzeddin Gur, Semih Yavuz, Yu Su, and Xifeng Yan. Dialsql: Dialogue based structured query generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1339–1349. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-1124.

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. Training classifiers with natural language explanations. *CoRR*, abs/1805.03818, 2018. URL http://arxiv.org/abs/1805.03818.

Michael Heilman and Noah A. Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010.

Huang Hu, Xianchao Wu, Bingfeng Luo, Chongyang Tao, Can Xu, Wei Wu, and Zhan Chen. Playing 20 Question Game with Policy-Based Reinforcement Learning. *arXiv e-prints*, 2018.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. *CoRR*, abs/1704.08760, 2017. URL http://arxiv.org/abs/1704.08760.

J. F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Trans. Inf. Syst.*, 2(1):26–41, January 1984. ISSN 1046-8188. doi: 10.1145/357417.357420. URL http://doi.acm.org/10.1145/357417.357420.

Sang-Woo Lee, Yu-Jung Heo, and Byoung-Tak Zhang. Answerer in Questioner's Mind for Goal-Oriented Visual Dialogue. 2018. URL https://arxiv.org/pdf/arXiv:1802.03881v3.pdf.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi S. Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluís Màrquez i Villodre. Denoising bodies to titles: Retrieving similar questions with recurrent convolutional models. *CoRR*, abs/1512.05726, 2015. URL http://arxiv.org/abs/1512.05726.

Tao Lei, Yu Zhang, Sida I. Wang, Hui Dai, and Yoav Artzi. Simple recurrent units for highly parallelizable recurrence. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

Jiwei Li, Alexander H. Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. Dialogue learning with human-in-the-loop. *CoRR*, abs/1611.09823, 2016. URL http://arxiv.org/abs/1611.09823.

Charles X. Ling, Qiang Yang, Jianning Wang, and Shichao Zhang. Decision trees with minimal costs. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pp. 69–, New York, NY, USA, 2004. ACM. ISBN 1-58113-838-5. doi: 10.1145/1015330.1015369. URL http://doi.acm.org/10.1145/1015330.1015369.

Ryan Lowe, Nissan Pow, Iulian V. Serban, and Joelle Pineau. The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. *Arxiv*, (September): 285–294, 2015.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016. URL http://arxiv.org/abs/1606.05250.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018.

Siva Reddy, Danqi Chen, and Christopher D Manning. CoQA: A conversational question answering challenge. *Transactions of the Association of Computational Linguistics (TACL)*, 2019.

Scott Reed, Zeynep Akata, Bernt Schiele, and Honglak Lee. Learning deep representations of fine-grained visual descriptions. In *IEEE Computer Vision and Pattern Recognition*, 2016.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009. ISSN 1554-0669. doi: 10.1561/1500000019. URL http://dx.doi.org/10.1561/1500000019.

Darsh J Shah, Tao Lei, Alessandro Moschitti, Salvatore Romeo, and Preslav Nakov. Adversarial Domain Adaptation for Duplicate Question Detection. 9 2018. URL http://arxiv.org/abs/1809.02255.

Paul E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4(2):161–186, Nov 1989. ISSN 1573-0565. doi: 10.1023/A:1022699900025. URL https://doi.org/10.1023/A:1022699900025.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Mengqiu Wang and Christopher D Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010.

Sida I. Wang, Percy Liang, and Christoper D. Manning. Learning language games through interaction, 2016.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system, 2016.

Xianchao Wu, Huang Hu, Momo Klyen, Kyohei Tomita, and Zhan Chen. Q20: Rinna riddles your mind by asking 20 questions. *Japan NLP*, 2018.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 2017 Annual Meeting of the Association for Computational Linguistics*, 2017. URL https://arxiv.org/pdf/1612.01627.pdf.

Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. Diverse few-shot text classification with multiple metrics, 2018.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. Multi-view response selection for human-computer conversation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016. URL https://aclweb.org/anthology/D16-1036.

Xiangyang Zhou, Lu Li, Daxiang Dong, Yi Liu, Ying Chen, Wayne Xin Zhao, Dianhai Yu, and Hua Wu. Multi-turn response selection for chatbots with deep attention matching network. In *Proceedings of International Conference on Computational Linguistics*, 2018. URL http://www.aclweb.org/anthology/P18-1103.

# A APPENDICES

## A.1 DATA COLLECTION

**Query collection qualification** One main challenge for the collection process lies within familiarizing the workers with the set of target documents. To make sure we get good quality annotation, we set up a two-step qualification task. The first one is to write paraphrase with complete information. After that, we reduce the number of workers down to $50$. These workers then generate $19,728$ paraphrase queries. During the process, the workers familiarize themselves with the set of documents. We then post the second task (two rounds), where the workers try to provide initial queries with possibly insufficient information. We select $25$ workers after the second qualification task and collect $3,831$ initial queries for the second task.

**Attribute Collection Qualification** To ensure the quality of target-tag annotation, we use the pre-trained model to rank-order the tags and pick out the highest ranked tags (as positives) and the lowest ranked tags (as negatives) for each target. The worker sees in total ten tags without knowing which ones are the negatives. To pass the qualifier, the workers need to complete annotation on three targets without selecting any of the negative tags.

| | **Tag Ranks** | | | | |
|---|---|---|---|---|---|
| | 1-10 | 11-20 | 21-30 | 31-40 | 41-50 |
| Mean # of tags | 3.31 | 1.45 | 0.98 | 0.61 | 0.48 |
| N.A. (%) | 1.9 | 30.7 | 43.6 | 62.1 | 65.2 |
| Mean $\kappa$ | 0.62 | 0.54 | 0.53 | 0.61 | 0.61 |

Table A.1: Target-tag annotation statistics. We show five sets of tags to the annotators. The higher ranked ones are more likely to be related to the given target. The row mean # tags is the mean number of tags that are annotated to a target, N.A. is the percentage of the tasks are annotated as *"none of the above"*, and mean $\kappa$ is the mean pairwise Cohen's $\kappa$ score.

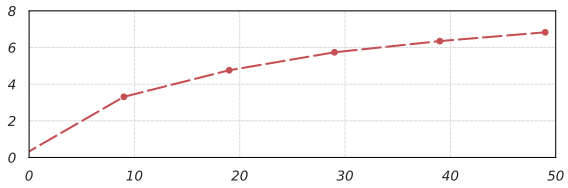


Figure A.1: Accumulated number of tags assigned to the targets (y-axis) by AMT against tag ranking (x-axis). The ranking indicates the relevance of the target-tag pairs from the pre-trained model. The curve plateaued at rank $50$ suggests that the lower ranked tags are less likely to be assigned to the target by the crowdsourcing workers.

## A.2 IMPLEMENTATION DETAILS

**Learning Components** Here we describe the detailed implementation of the text encoder and the policy controller network. We use a single-layer bidirectional Simple Recurrent Unit (SRU) as the encoder for the FAQ suggestion task and two layer bidirectional SRU for bird identification task. The encoder uses pre-trained fastText Bojanowski et al. (2016) word embedding of size $300$ (fixed during training), hidden size $150$, batch size $200$, and dropout rate $0.1$. The policy controller is a two layer feed-forward network with hidden layer size of $32$ and ReLU activation function. We use Noam learning rate scheduler with initial learning rate $1e-3$, warm-up step $4,000$ and Noam scaling factor $2.0$. The policy controller is a $2$ layer feed-forward network with a hidden layer of $32$ dimensions and ReLU activation. The network takes the current step and the top-k values of belief probabilities as input. We choose $k = 20$ and allow a maximum of $10$ interaction turns.

## A.3 ANALYSIS

**Text Encoder Training** We use initial queries as well as paraphrase queries to train the encoder, which has around $16$K target-query examples. The breakdown analysis is shown in Table A.2. To

| Text Input | | Init Query | | Init Query + Tags | | Init + Paraphrase Query | | Full Model | |
|---|---|---|---|---|---|---|---|---|---|
| init query | tags | Acc@1 | Acc@3 | Acc@1 | Acc@3 | Acc@1 | Acc@3 | Acc@1 | Acc@3 |
| ✓ | ✗ | 0.28 | 0.47 | 0.32 | 0.51 | 0.35 | 0.60 | 0.38 | 0.61 |
| ✗ | ✓ | 0.31 | 0.50 | 0.57 | 0.79 | 0.56 | 0.74 | 0.70 | 0.87 |
| ✓ | ✓ | 0.36 | 0.58 | 0.55 | 0.79 | 0.63 | 0.81 | 0.76 | 0.91 |

Table A.2: Comparison of the suggestion modules trained with different training data. Each model is evaluated on three different tasks. First, use initial queries to predict targets. Second, use all attributes tags to predict targets; third, use both initial queries and tags as text input to predict targets. Each model is evaluated on Accuracy@$\{1, 3\}$.

see the effectiveness of the tag in addition to initial query, we generate pseudo-queries by combining existing queries with sampled subset of tags from the targets. This augmentation strategy is shown to be useful to improve the classification performance.

**Policy Controller Learning** Finally, Figure A.2 shows the learning curves of our model with the policy controller trained with different turn penalty $r_a \in \{-0.5, -1, -3\}$. We observe interesting exploration behaviors during the first $1,000$ training episodes, shown in the middle and right plots of Figure A.2. The models achieve relatively stable recall numbers after the early exploration stage. As expected, the three runs converge to using different number of expected turns due to the choice of different $r_a$ values.

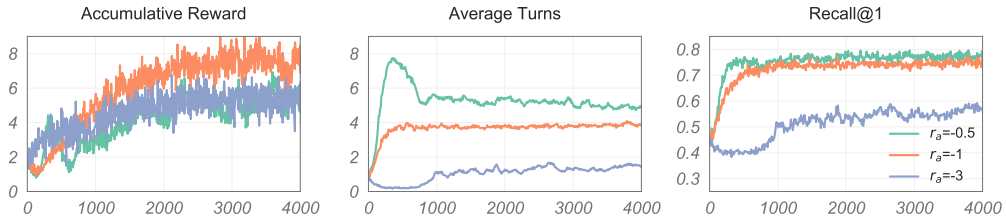


Figure A.2: Learning curves of our full model. We show accumulative reward (left), interaction turns (middle), and Accuracy@1 (right) on the test set, where x-axis is the number of episodes run (400 trials per episode). The results are compared on different turn penalty $r_a$.

### A.4 Human Evaluation

Each interaction session starts with presenting the annotator an user scenario (e.g a bird image or an issue with your phone). The annotator inputs an initial query accordingly and then answers follow-up questions selected by the system.

**FAQ Suggestion** We evaluate prediction accuracy, system rationality, and the number of counts by letting the system interact with human judges. We design user scenario for each target to present to the worker. At the end of each interaction, the predicted FAQ and the ground truth will be presented to the user as shown in the top right panel in Figure A.3. The user needs to answer the following questions: "*How natural is the interaction?*" and "*Do you feel understood by the system during the interactions?*" on the scale of $-2$ (strongly disagree) to 2 (strongly agree), which we record as naturalness and rationality in Table A.4. Our full model performs best on Accuracy@1, naturalness, and rationality. We show human evaluation examples in Table A.3.

**Bird Identification** The interface for bird identification task is similar to the FAQ suggestion task. Instead of presenting a scenario, we show a bird image to the user. The user needs to describe the bird to find out its category, which is analogous to writing an initial query. We allow the user to reply "not visible" if part of the bird is hidden or occluded. With such reply, the system stops asking attributes from the same label group. At the end of the interaction, the predicted and ground-truth bird images along with their categories are presented to the user as shown in the bottom right panel in Figure A.3. Again, the user needs to fill out a similar questionnaire as in FAQ suggestion task. The bird identification task is very challenging due to its fine-grained categories, where many bird images look almost identical while belonging to different classes. Our full system improves Accuracy@1 from $20\%$ to $45\%$ against non-interactive baselines after less than 4 turns of interaction. Besides

| | |
|---|---|
| U: | How can I text my friends house phone? |
| S: | Text message to landline? |
| U: | Yes. |
| S: | Do you want to send or receive text? |
| U: | Yes. |
| S: | Please read: *How do I send a text message to landline?* |
| U: | I'm in sprint's one up and want to upgrade my phone. |
| S: | Do you want annual upgrade option? |
| U: | Yes. |
| S: | Do you want to exercise an option? |
| U: | Yes. |
| S: | Please read: *I am enrolled in Sprint One Up. How do I exercise my annual upgrade option?* |
| U: | How does hotspot affect battery. |
| S: | Are you using mobile hotspot? |
| U: | Yes. |
| S: | Please read: *Does using my phone as a mobile Wi-Fi hotspot drain my battery?* |

Table A.3: Examples of user interactions for FAQ suggestion human evaluation.

| Dataset | Models | Count | Initial Acc@1 | Final Acc@1 | Naturalness | Rationality |
|---|---|---|---|---|---|---|
| FAQ | OURS | 142 | 30% | 64% | 0.4 | 0.25 |
| | OURS(Fixed Turn) | 43 | 30% | 54% | -0.17 | -0.14 |
| | STATIC INTERACT | 29 | 28% | 35% | 0.03 | -0.03 |
| CUB | OURS | 48 | 21% | 45% | 0.21 | 0.60 |
| | OURS(Fixed Turn) | 24 | 13% | 26% | -0.50 | -0.60 |
| | STATIC INTERACT | 29 | 25% | 29% | 0.1 0 | -0.20 |

Table A.4: Human evaluation results on FAQ and CUB dataset on our proposed model and several baslines. The three FAQ systems ask 2.8, 3 and 3 turns of questions, respectively. The three CUB systems ask 3.3, 4 and 4 turns of questions. The system is evaluated with both on performance and user experience. Performance include the initial and final Accuracy@1. The user experience score include both naturalness and rationality for both task. We also add human rated correctness for bird identification task.

Accuracy@1, we also study cases where human judges consider the predicted image to be almost identical to the true image even if the predicted class is incorrect. For this human rated Accuracy@1 (correctness), our system reaches to $75\%$. To better understand the task and the model behavior, we show the confusion matrix of the final model prediction after interaction in Figure A.4. In the 200 bird classes, there are 21 different kinds of sparrows and 25 different warbler. Those fine-grained bird classes identification induces most model errors. Figure A.5 show how the confusion matrix change, adding the interactions. The model makes improvement in distinct and also similar bird types.

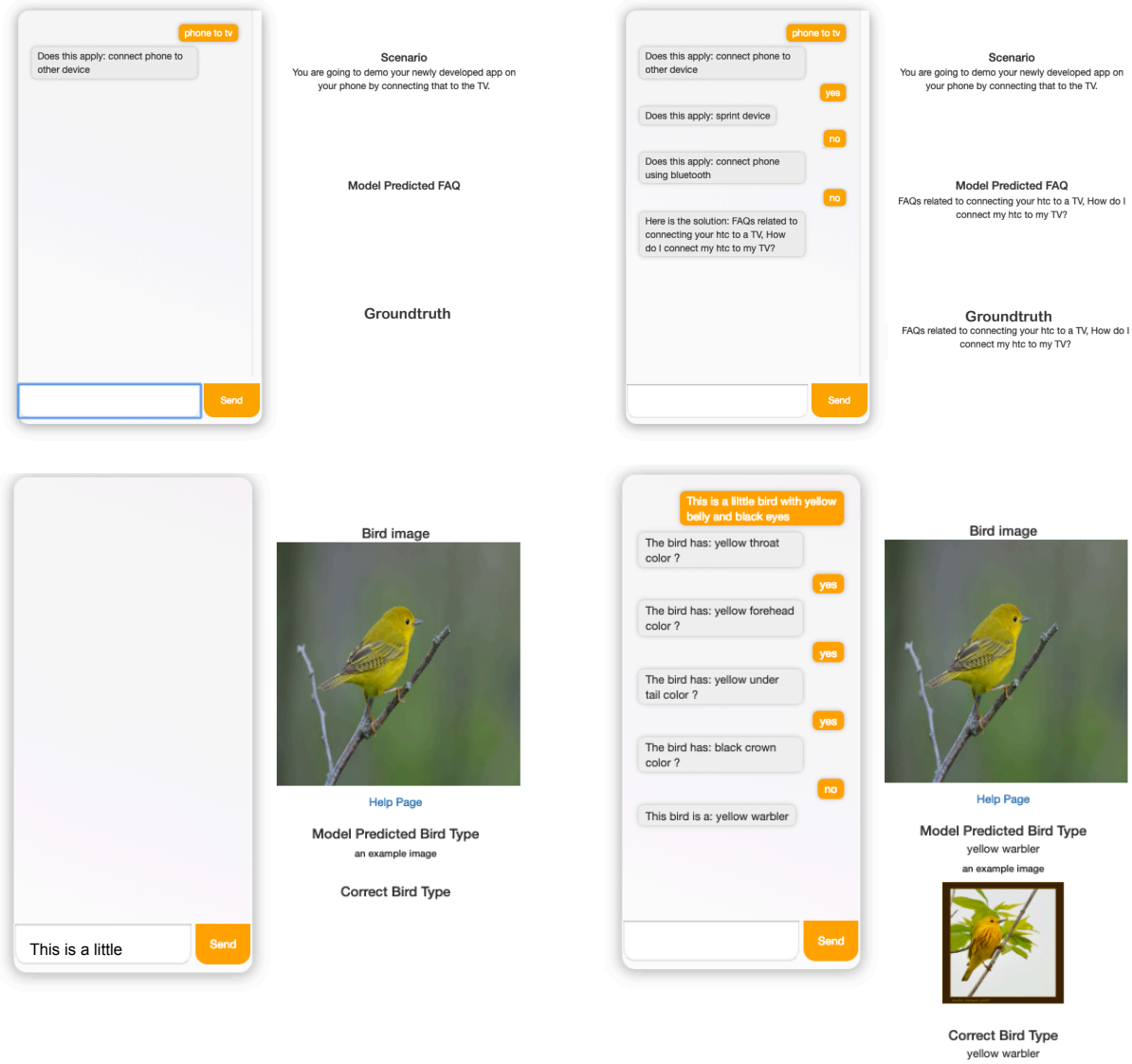


Figure A.3: User interface for FAQ suggestion task (top) and bird identification (bottom) tasks. Left panel shows the interface at the beginning of the interaction and the right panel shows the inter face at the end of the interaction

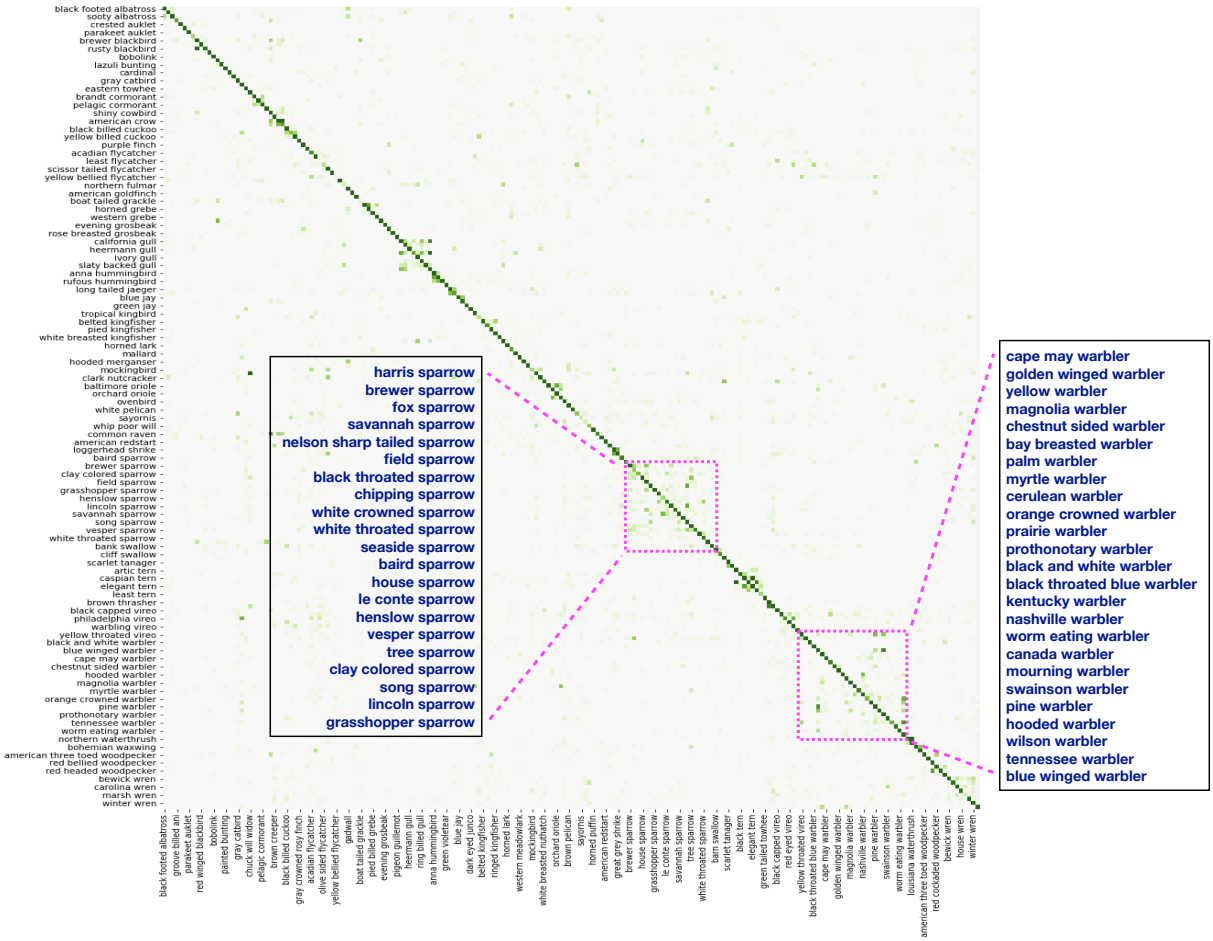


Figure A.4: Confusion matrix of our final output for bird identification task.

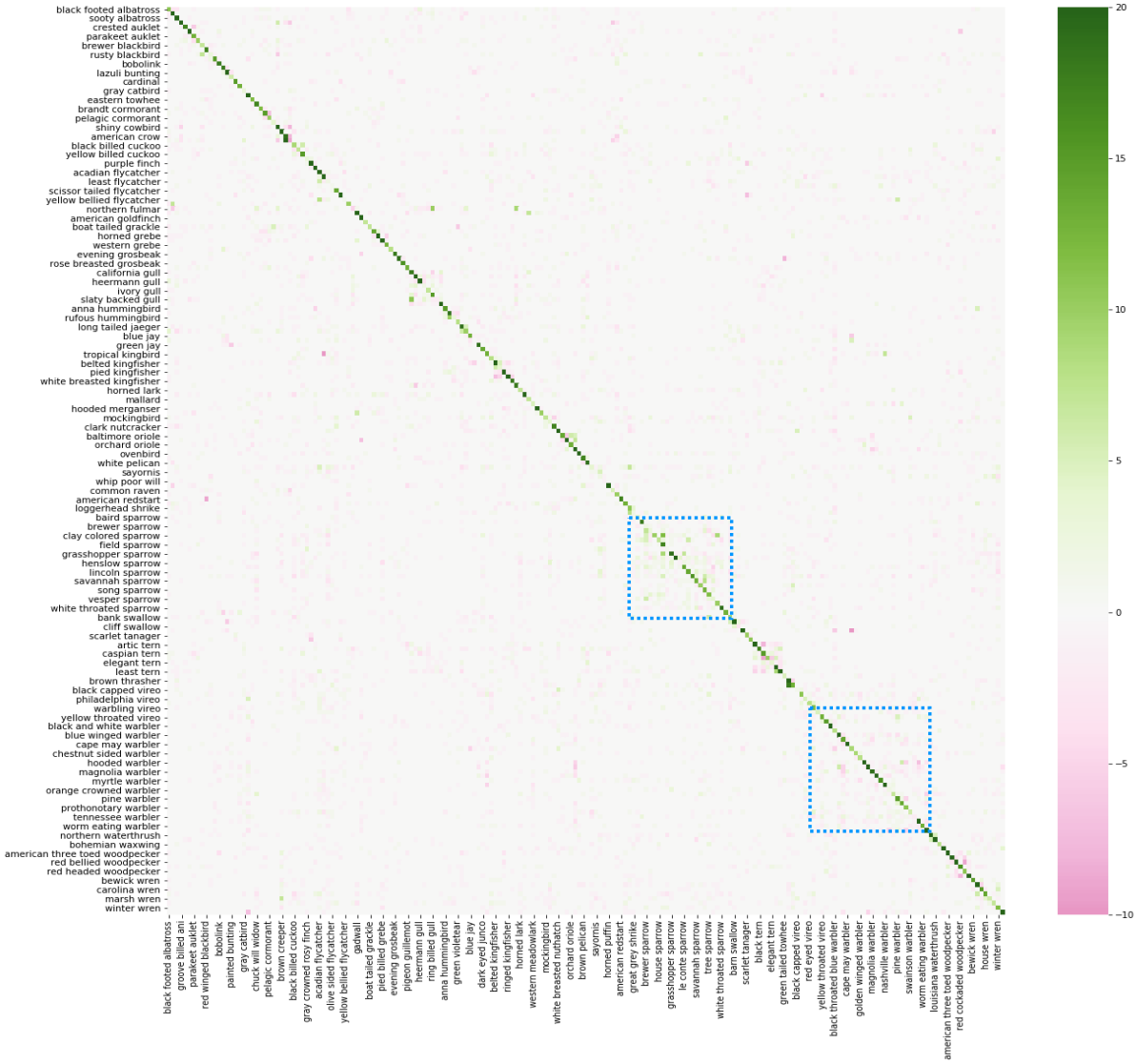


Figure A.5: Confusion matrix difference between the initial query with and without the interactions. We desire high value in the diagonal part and low value elsewhere.