INTERACTIVE CLASSIFICATION BY ASKING INFORMATIVE QUESTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Natural language systems often rely on a single, potentially ambiguous input to make one final prediction, which may simplify the problem but degrade end user experience. Instead of making predictions with the natural language query only, we ask the user for additional information using a small number of binary and multiple-choice questions in order to better help users accomplish their goals while minimizing their effort. At each turn, our system decides between asking the most informative question or making the final classification prediction. The approach enables bootstrapping the system using simple crowdsourcing annotations without expensive human-to-human interaction data. Evaluation demonstrates that our method substantially increases classification accuracy, while effectively balancing the number of questions with the improvement to final accuracy.

1 INTRODUCTION

Responding to natural language queries through simple, single-step classification has been studied extensively in many applications, including user intent prediction (Chen et al., 2019; Qu et al., 2019), and information retrieval (Kang & Kim, 2003; Rose & Levinson, 2004). Typical methods rely on a single user input to produce an output, missing an opportunity to interact with the user to reduce ambiguity and improve the final prediction. For example, users may under-specify a request due to incomplete understanding of the domain; or the system may fail to correctly interpret the nuances of the input query. In both cases, a low quality input could be mitigated by further interaction with the user.

In this paper we propose a simple but effective interaction paradigm that consists of a sequence of binary and multiple choice questions allowing the system to ask the user for more information. Figure 1 illustrates the types of interaction supported by this method, showcasing the opportunity for clarification while avoiding much of the complexity involved in unrestricted natural language interactions. Following a natural language query from the user, our system then decides between posing another question to obtain more information or finalizing the current prediction. Unlike previous work which assumes access to full interaction data (Wu et al., 2018; Hu et al., 2018; Lee et al., 2018; Rao & Daumé III, 2018), we are interested in bootstrapping an interaction system using simple and relatively little annotation effort. This is particularly important in real-world applications, such as in virtual assistants, where the supported classification labels are subject to change and thereby require a lot of re-annotation.

We propose an effective approach designed for interaction efficiency and simple system bootstrapping. Our approach adopts a Bayesian decomposition of the posterior distributions over classification labels and user's responses through the interaction process. Due to the decomposition, we can efficiently compute and select the next question that provides the maximal expected information based on the posteriors. To further balance the potential increase in accuracy with the cost of asking additional questions, we train a policy controller to decide whether to ask additional questions or return a final prediction. Our method also enables separately collecting natural language annotations to model the distributions of class labels and user responses. Specifically, we crowdsource initial natural language queries and question-answer pairs for each class label, alleviating the need for Wizard-of-Oz style dialog annotations (Kelley, 1984; Wen et al., 2016). Furthermore, we leverage the natural language descriptions of class labels, questions and answers to help estimate their correlation and reduce the need for heavy annotation.

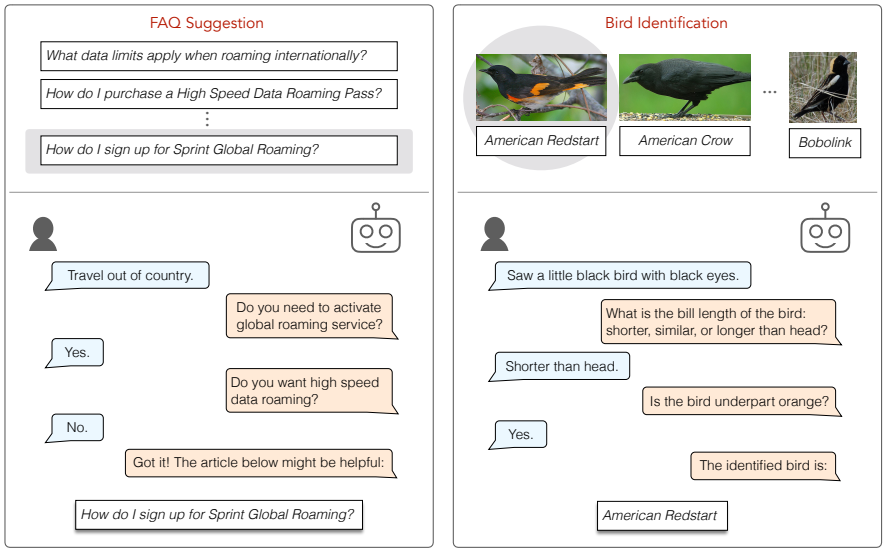

Figure 1: Two example use cases of interactive classification system: providing customer the best trouble-shooting suggestion (left) and helping user identify bird species from text interactions (right). The top parts show example classification labels: FAQ documents or bird species, where the ground truth label of each interaction example is shaded. The lower parts show how a user interact with the system typically. The user starts with an initial natural language query. At each step, the system asks a clarification question. The interaction ends when the system returns an output label.

We evaluate our method on two public tasks: FAQ suggestion (Shah et al., 2018) and bird identification using the text and attribute annotations of the Caltech-UCSD Birds dataset (Wah et al., 2011). The first task represents a virtual assistant application in a trouble-shooting domain, while the second task provides well-defined multiple-choice question annotations and naturally noisy language inputs. Our experiments show that adding user interactions significantly increases the classification accuracy, when evaluating against both a simulator and a real human. With one clarification question, our system obtains a relative accuracy boost of 40% and 65% for FAQ suggestion and bird identification compared to no-interaction baselines on simulated evaluation. Given at most five turns of interaction, our approach improves accuracy by over 100% on both tasks in both simulated and human evaluation.

2 Technical Overview

Notation Our goal is to classify a natural language query as one class label through interactions. To this end, we treat the classification label y, interaction question q as well as the user response r as random variables. We denote one possible value or assignment of the random variable using subscripts, such as $y = y_i$ and $q = q_j$. We use superscripts for the observed value of the random variable at a given time step, for example, q^t is a question asked at time step t. Whenever it is clear from the context, we simply write y_i as the short notation of $y = y_i$. For example, $p(r|q_j, y_i)$ denotes the conditional distribution of r given $y = y_i$ and $q = q_j$, and $p(r_k|q_j, y_i)$ further specifies the corresponding probability when $r = r_k$.

An interaction starts with the user providing an initial user query x. At each turn t, the system can choose to select a question q^t, to which the user respond with r^t. We consider two types of questions in this work: binary and multiple choice questions. The predefined set of possible answers for a question q^t is $\mathcal{R}(q^t)$, where $\mathcal{R}(q^t)$ is $\{yes, no\}$ for binary questions, or a predefined set of question-specific values for multiple choice questions. We denote an interaction up to time t as $X^t = (x, (q^1, r^1), \ldots, (q^t, r^t))$, and the set of possible class labels as $\mathcal{Y} = \{y_1, \ldots, y_N\}$. Figure 1 shows exemplary interactions in our two evaluation domains.

Model We model the interactive process using a parameterized distribution over class labels that is conditioned on the observed interaction (Section 4.1), a question selection criterion (Section 4.2),

and a parameterized policy controller (Section 4.5). At each time step t, we compute the belief of each $y_i \in \mathcal{Y}$ conditioned on X^{t-1}. The trained policy controller decides between two actions: to return the current best possible label or to obtain additional information by asking a question. The model selects the question that maximizes the information gain. After receiving the user response, the model updates the beliefs over the classification labels.

Learning We use crowdsourced non-interactive data to bootstrap model learning. The crowdsourced data collection consists of two sub-tasks. First, we obtain a set of user initial queries $\mathcal{X}_i$ for each label y_i. For example, for an FAQ, 'How do I sign up for Spring Global Roaming', an annotator can come up with an initial query as 'Travel out of country'. Second, we ask annotators to assign text tags to each y_i, and convert these tags into a set of question-answer pairs $\mathcal{A}_i = \{(q_m, r_m)\}_{m=1}^{M_i}$. Here q_m denotes a templated question and r_m denotes the answer. For example, a question 'What is your phone operating system?' can pair with one of the following answers: 'IOS', 'Android operating system', 'Windows operating system' or 'Not applicable'. We denote this dataset as $\{(y_i, \mathcal{X}_i, \mathcal{A}_i)\}_{i=1}^{N}$. We describe the data collection process in Section 5. We use this data to train our text embedding model (Section 4.3), to create a user simulator (Section 4.4), and to train the policy controller to minimize the number of interaction turns while achieving high classification accuracy (Section 4.5).

Evaluation We report classification accuracy of the model, and study the trade-off between accuracy and the number of the turns that the system takes. We run our system against a user simulator and real human users. When performing human evaluation, we additionally collect qualitative ratings of the interactions.

3 Related Work

Learning from Feedback A lot of recent work have leveraged human feedback to train natural language processing models, including dialogue learning (Li et al., 2016), semantic parsing (Artzi & Zettlemoyer, 2011; Wang et al., 2016; Iyer et al., 2017) and text classification (Hancock et al., 2018). These works collect user feedback after the model-predicting stage and treat user feedback as extra offline training data to improve the model. In contrast, our model leverages the user interaction and makes model prediction accordingly in an online fashion. Human feedback has been incorporated in reinforcement learning (Christiano et al., 2017; Nguyen et al., 2018) as well. For instance, (Christiano et al., 2017) learns a reward function from human preferences to provide richer rewards and (Nguyen et al., 2018) uses language-illustrated subgoals as indirect interventions instead of conventional expert demonstrations.

Modeling Interaction Language-based interaction has recently attracted a lot of attention in areas such as visual question answering (De Vries et al., 2016; Lee et al., 2018; Chattopadhyay et al., 2017; Das et al., 2017; Lee et al., 2019; Shukla et al., 2019), SQL generation (Gur et al., 2018; Yao et al., 2019), information retrieval (Chung et al., 2018; Aliannejadi et al., 2019) and multi-turn text-based question answering (Rao & Daumé III, 2018; Reddy et al., 2019; Choi et al., 2018). Many of these works require learning from full-fledged dialogues (Wu et al., 2018; Hu et al., 2018; Lee et al., 2018; Rao & Daumé III, 2018) or conducting Wizard-of-Oz dialog annotations (Kelley, 1984; Wen et al., 2016). Instead of utilizing unrestricted but expensive conversations, we limit ourselves to a simplified type of interaction consisting of multiple-choice and binary questions. This allows us to reduce the complexity of data annotation while still achieving efficient interaction and high accuracy.

Our question selection method is closely related to the prior work of (Lee et al., 2018; Rao & Daumé III, 2018; Kovashka & Grauman, 2013; Ferecatu & Geman, 2007; Guo et al., 2018). For instance, (Kovashka & Grauman, 2013) refine image search by asking to compare visual qualities against selected reference images, and (Lee et al., 2018) perform object identification in image by posing binary questions about the object or its location. Both works and our system use an entropy reduction criterion to select the best question. Our work makes use of a Bayesian decomposition of the joint distribution and can be easily extended to other model-driven selection. We also highlight the modeling of natural language to estimate information and probabilities. More recently, (Rao & Daumé III, 2018) proposes a learning-to-ask model by modeling the expected utility obtained by a question. Our selection method can be considered as a special case when entropy is used as the utility. In contrast to (Rao & Daumé III, 2018), our work models the entire interaction history instead

of a single turn of follow-up questioning. Our model is trained using crowdsourced annotations, while (Rao & Daumé III, 2018) uses real user-user interaction data.

Learning 20Q Game Our task can be viewed as an instance of the popular 20-question game (20Q), which has been applied to the knowledge base of celebrities (Chen et al., 2018; Hu et al., 2018). Our work differs in two fold. First, our method models the natural language descriptions of classification targets, quesitons ans answers, instead of treating them as categorical or structural data as in knowledge base. Second, instead of focusing on the 20Q game (on celebrities) itself, we aim to help users accomplish realistic goals with minimal interaction effort.

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, decide if to ask the question or return the classification output using a policy controller and select a question to ask using information gain if needed.

4.1 BELIEF PROBABILITY DECOMPOSITION

We decompose the conditional probability $p(y = y_i \mid X^t)$ using Bayes rule:

$$\begin{aligned} p(y = y_i \mid X^t) &= p(y_i \mid X^{t-1}, q^t, r^t) \\ &\propto p(r^t, q^t, y_i \mid X^{t-1}) \\ &= p(r^t \mid q^t, y_i, X^{t-1}) \cdot p(q^t \mid y_i, X^{t-1}) \cdot p(y_i \mid X^{t-1}) \ . \end{aligned} \tag{1}$$

We make two simplifying assumptions for Eq. (1). First, we assume the user response only depends on the provided question q^t and the underlying target label y_i, and is independent of past interactions. The assumption simplifies $p(r^t \mid q^t, y_i, X^{t-1})$ as $p((r^t \mid q^t, y_i)$. Second, we deterministically select the next question q^t given the interaction history X^{t-1} (described in Section 4.2). As a result, $p(q = q^t \mid y_i, X^{t-1}) = 1$ and otherwise would be zero if $q \neq q^t$. These two assumptions allow us to rewrite the decomposition as:

$$p(y = y_i \mid X^t) \ \propto p(r^t \mid q^t, y_i) \cdot 1 \cdot p(y_i \mid X^{t-1}) = p(y_i \mid x) \ \prod_{\tau=1}^{t} p(r^\tau \mid q^\tau, y_i); . \tag{2}$$

That is, predicting the classification label given the observed interaction X^t can be reduced to modeling $p(y_i \mid x)$ and $p(r_k \mid q_j, y_i)$, i.e. the label probability given the initial query only and the probability of user response conditioned on the chosen question and class label. This factorization enables us to leverage separate annotations to learn these two components directly, alleviating the need for collecting costly full user interactions.

4.2 QUESTION SELECTION USING INFORMATION GAIN

The system selects the question q^t to ask at turn t to maximize the efficiency of the interaction. We use a maximum information gain criterion. Given X^{t-1}, we compute the information gain on classification label y as the decrease on entropy by observing possible answers to question q:

$$IG(y\,;\, q \mid X^{t-1}) = H(y \mid X^{t-1}) - H(y \mid X^{t-1}, q) \ ,$$

where $H(\cdot \mid \cdot)$ denotes the conditional entropy. Intuitively, the information gain measures the amount of information obtained about the variable y by observing the value of another variable q. Because the first entropy term $H(y \mid X^{t-1})$ is a constant regardless of the choice of q, the selection of q^t is equivalent to

$$q^t = \arg\min_{q_j} H(y \mid X^{t-1}, q_j) = \arg\min_{q_j} \sum_{r_k \in \mathcal{R}(q_j)} p(r_k \mid X^{t-1}, q_j) \cdot H(y \mid X^{t-1}, q_j, r_k) \ \ ,$$

where

$$p(r_k \mid X^{t-1}, q_j) = \sum_{y_i \in \mathcal{Y}} p(r_k, y_i \mid X^{t-1}, q_j) = \sum_{y_i \in \mathcal{Y}} p(r_k \mid q_j, y_i) \cdot p(y_i \mid X^{t-1}) \ \ ,$$

$$H(y \mid X^{t-1}, q_j, r_k) = \sum_{y_i \in \mathcal{Y}} p(y_i \mid X^{t-1}, q_j, r_k) \cdot \log p(y_i \mid X^{t-1}, q_j, r_k) \ .$$

Here we use the independent assumption stated in Section 4.1 to calculate $p(r_k \mid X^{t-1}, q_j)$. Both $p(r_k \mid X^{t-1}, q_j)$ and $p(y_i \mid X^{t-1}, q_j, r_k)$ can be iteratively updated utilizing $p(y_i \mid x)$ and $p(r_k \mid q_j, y_i)$ as the interaction progresses (Eq. 2), resulting in efficient computation of information gain.

4.3 Modeling the Distributions

We model $p(y_i \mid x)$ and $p(r_k \mid q_j, y_i)$ by encoding the natural language descriptions of questions, answers and classification labels[1]. That is, we do not simply treat the labels, questions and answers as categorical variables in our model. Instead, we leverage their natural language aspects to better estimate their correlation, reduce the need for heavy annotation and improve our model in low-resource (and zero-shot) scenarios. Specifically, we use a shared neural encoder $\mathbf{enc}(\cdot)$ to encode all texts. Both probability distributions are computed using the dot-product score, i.e. $S(u, v) = \mathbf{enc}(u)^\top \mathbf{enc}(v)$ where u and v are two pieces of text.

The probability of predicting the label y_i given an initial query x is:

$$p(y_i \mid x) = \frac{\exp(S(y_i, x))}{\sum_{y_j \in \mathcal{Y}} \exp(S(y_j, x))} \ .$$

The probability of an answer r_k given a question q_j and label y_i is a linear combination of the observed empirical probability $\hat{p}(r_k \mid q_j, y_i)$ and a parameterized estimation $\tilde{p}(r_k \mid q_j, y_i)$:

$$p(r_k \mid q_j, y_i) = \lambda \cdot \hat{p}(r_k \mid q_j, y_i) + (1 - \lambda) \cdot \tilde{p}(r_k \mid q_j, y_i) \ ,$$

where $\lambda \in [0, 1]$ is a hyper-parameter. We use the question-answer annotations $\mathcal{A}_i$ for each label y_i to estimate $\hat{p}(r_k \mid q_j, y_i)$ using the empirical count. For example, in the FAQ suggestion task, we collect multiple user responses for each question and class label, and average across annotator to estimate $\hat{p}$ (Section 5). The second term $\tilde{p}(r_k \mid q_j, y_i)$ is estimated using texts:

$$\tilde{p}(r_k \mid q_j, y_i) = \frac{\exp(w \cdot S(q_j \# r_k, y_i) + b)}{\sum_{r_l \in \mathcal{R}(q_j)} \exp(w \cdot S(q_j \# r_l, y_i) + b)} \ ,$$

where $q_j \# r_k$ is a concatenation of the question q_j and the answer r_k[2] and $w, b \in \mathbb{R}$ are scalar parameters. Because we do not collect complete annotations to cover every label-question pair, $\tilde{p}$ provides a smoothing of the partially observed counts using the learned encoding $S(\cdot)$.

We estimate the $\mathbf{enc}(\cdot)$ with parameters ψ by pre-training using the dataset, $\{(y_i, \mathcal{X}_i, \mathcal{A}_i)\}_{i=1}^N$, as described earlier. 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, y_i) with all its initial queries $x \in \mathcal{X}_i$. We also create $(q_m \# r_m, y_i)$ for each question-answer pair (q_m, r_m) annotated with the label y_i. We perform gradient descent to minimize the cross-entropy loss:

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

The second term requires summation over all v', 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. The parameters ψ, w and b are fine-tuned using reinforcement learning during training of the policy controller (Section 4.5).

4.4 User Simulator

The user simulator provides initial queries to the system, responds to the system initiated clarification questions and judges if the returned label is correct. The simulator is based on held-out dataset

[1] The text representation of a label is the FAQ document or the bird name for instance.

[2] For example, for a templated question 'What is your phone operating system?' and an answer 'IOS', q_m = 'phone operating system' and r_m = 'IOS', therefore, $q_m \# r_m$ = 'phone operating system IOS'.

Algorithm 1: Full model training

Initialize text encoder, model for $p(y \mid x)$, and user simulator SIM
for *episode = 1 .. M* **do**
  Sample $(x, \hat{y})$ 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**
      $q^t = \arg\max_{q_j \in \mathcal{Q}} \; IG(y\,;\, q_j \mid X^{t-1})$
      $r^t \sim p'(r \mid q^t, \hat{y})$
  end
  $y^* = \arg\max_{y_i} \; p(y_i \mid X^{t-1})$
  reward $=$ SIM$(y^*, \hat{y})$
  Update w, b, θ using policy gradient
end

$\{(y_i, \mathcal{X}'_i, \mathcal{A}'_i\}_{i=1}^{N}$ of tuples of a goal y_i, a set of initial queries $\mathcal{X}'_i$, and a set of question answer pairs $\mathcal{A}'_i = \{(q_m, r_m)\}_{m=1}^{M'_i}$. We estimate the simulator response distribution $p'(r_k \mid q_j, y_i)$ using smoothed empirical counts from the data. While this data is identical in structure to our training data, we keep it separated from the data used to estimate $S(\cdot)$, $p(y_i \mid x)$ and $p(r_k \mid q_j, y_i)$ (Section 4.3).

At the beginning of an interaction, the simulator selects a target label $\hat{y}$, and samples a query x from the associated query set 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 distribution $p'(r \mid q^t, \hat{y})$. Sampling provides natural noise to the interaction, and our model has no knowledge of 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 a 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 most probable classification label. The policy controller $f(\cdot, \cdot; \theta)$ is a feed-forward network parameterized by θ 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 label y_i with highest probability is returned using $\arg\max_{y_i \in \mathcal{Y}} p(y_i \mid X^{t-1})$.

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 $p(r_k \mid q_j, y_i)$ by back-propagating through the policy gradient. We keep the $\mathbf{enc}(\cdot)$ parameters fixed during this process.

5 Data Collection

We design a crowdsourcing process to collect data for the FAQ task using Amazon Mechanical Turk[3]. 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 that they would provide to such a system. The set of initial queries that is collected for each document

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

y_i is $\mathcal{X}_i$. We encourage workers to provide incomplete information and avoid writing a simple paraphrase of the FAQ. This results in more realistic and diverse utterances because users have limited knowledge of the system and the domain.

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 pre-defined ontology and can be a phrase or a single word, which describes the topic of the document. We heuristically remove duplicate tags to finalize the set. Next, experts heuristically combine some tags to categorical tags, while leave all the rest tags as binary tags. For example, tags 'IOS', 'Android operating system' and 'Windows operating system' are combined to form a categorical tag 'phone operating system'. We then use a small set of deterministic, heuristically-designed templates to convert tags into questions. For example, the tag 'international roaming' is converted into a binary question 'Is it about international roaming?'; the categorical tag 'phone operating system' is converted into a multi-choice question 'What is your phone operating system?'. Finally, we use non-experts to collect user responses to the questions of the FAQ. For binary questions, we ask workers to associate the tags to the FAQ target if they would respond 'yes' to the question. We show the workers a list of ten tags for a given target as well as '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. The workers do not engage in a multi-round interactive process. This allows for cheap and scalable collection.

6 Experimental Setup

Task I: FAQ Suggestion We use the FAQ dataset from (Shah et al., 2018). The dataset contains 517 troubleshooting documents by crawling Sprint's 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 learning the policy controller. We use the queries and tag annotations of the development and test documents for evaluation only. The classification targets contain all 517 documents during evaluation[4].

Task II: Bird Identification Our second set of experiments uses 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 tags, leaving us 8 categorical questions in total. The remaining 279 attributes are treated as binary tags and converted to binary questions. In addition, each image is annotated with 10 image captions describing the bird in the image (Reed et al., 2016). We use the image captions as initial user queries and bird species as labels. Since each image often contains only partial information about the bird species, the data is naturally noisy and provides challenging user interactions. We do not use the images from the dataset for model training. The images are only provided for grounding during human evaluation.

Baselines We compare our full model against the following baseline methods:

- No Interact: the best classification label is predicted using only the initial query. We consider two possible implementations: BM25, a common keyword-based scoring model for retrieval methods (Robertson & Zaragoza, 2009), and a neural model described in Section 4.3.
- Random Interact: at each turn, a random question is chosen and presented to the user. After T turns, the classification label is chosen according to the belief $p(y \mid X^T)$.

[4]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.

- Static Interact: questions are picked without conditioning on the initial user query using maximum information criterion. This is equivalent to using a static decision tree to pick the question, leading to the same first question, similar to (Utgoff, 1989; Ling et al., 2004).
- Variants of Ours: we consider several variants of our full model. First, we replace the policy controller with two termination strategies: one which ends interaction when $\max p(y \mid X^t)$ passes a threshold, and another one which ends interaction after the designated number of turns. Second, we disable the parameterized estimator $\tilde{p}(r_k \mid q_j, y_i)$ by setting λ to 1.

Evaluation We evaluate our model by running against both the user simulator and a real human. Given the user simulator, we evaluate the classification performance of our model and baselines using Accuracy@k, which is the percentage of time the correct target appears among the top-k predictions of the model. During human evaluation, we ask annotators to interact with our proposed or baseline models through a web-based interactive interface. Each interaction session starts with a user scenario[5] presented to an annotator (e.g a bird image or a device-troubleshooting scenario described in text). The annotator inputs an initial query accordingly and then answers follow-up questions selected by the system. Once the system returns its prediction, the annotator is asked to provide a few ratings of the interaction, such as *rationality – do you feel that you were understood by the system?*. We present more details of the human evaluation in Appendix A.3.

Implementation Details We use a fast recurrent neural network to encode texts (Lei et al., 2018). The policy controller receives three different rewards: a positive reward for returning the correct target ($r_p = 20$), a negative reward for providing the wrong target ($r_n = -10$) and a turn penalty for each question asked ($r_a = -1, \ldots, -5$). We report the averaged results over 3 independent runs for each model variant 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)	$38 \pm 0.5\%$	$61 \pm 0.3\%$	$23 \pm 0.1\%$	$41 \pm 0.2\%$
Random Interact	$39 \pm 0.3\%$	$62 \pm 0.4\%$	$25 \pm 0.1\%$	$44 \pm 0.1\%$
Static Interact	$46 \pm 0.5\%$	$66 \pm 0.6\%$	$29 \pm 0.2\%$	$50 \pm 0.3\%$
Ours	**79 ± 0.7%**	**86 ± 0.8 %**	**49 ± 0.3 %**	**69 ± 0.5 %**
w/ threshold	$72 \pm 0.6\%$	$82 \pm 0.7\%$	$41 \pm 0.3\%$	$59 \pm 0.4\%$
w/ fixed turn	$71 \pm 1.0\%$	$81 \pm 0.9\%$	$39 \pm 0.2\%$	$56 \pm 0.4\%$
w/ $\lambda = 1$	$66 \pm 0.8\%$	$71 \pm 1.0\%$	$40 \pm 0.1\%$	$56 \pm 0.2\%$

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. Best performances are in bold. We report the averaged results as well as the standard deviations from 3 independent runs for each model variant and baseline.

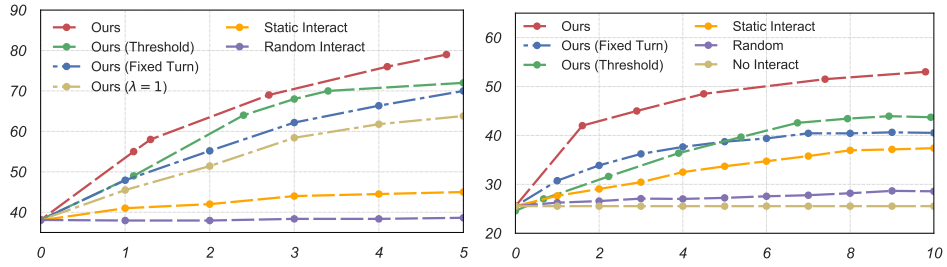

Figure 2: Accuracy@1 (y-axis) against turns of interactions (x-axis) for FAQ Suggestion (left) and Bird Identification (right) tasks

[5]Each scenario is related to a single groundtruth label and serves as the grounding of user interactions.

	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

Simulated 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 an Accuracy@1 of 38% for FAQ Suggestion and 23% for Bird Identification. The No Interact (BM25) baseline performs worst. The Random Interact baseline and the Static Interact baseline barely improve the performance, illustrating the challenge of building an effective interactive model. In contrast, our model and its variants obtain substantial gain in accuracy given a few turns of interaction. Our full model achieves an Accuracy@1 of 79% for FAQ Suggestion and 49% for Bird Identification using less than 5 turns, outperforming the No Interact (Neural) baseline by an absolute number of 41% and 26%. The two baselines with alternative termination strategies underperform the full model, indicating the effectiveness of the 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 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 the trade-off between classification accuracy and the number of the turns the system takes. The number of interactions changes as we vary the model termination strategy, which includes varying turn penalty r_a, the prediction threshold, and the predefined number of turns T. Our model with the policy controller or the threshold strategy does not explicitly control the number of turns, so we report the average number of turns across multiple runs for these two models. We achieve a relative accuracy boost of 40% for FAQ Suggestion and 65% for Bird Identification over no-interaction baselines with only one clarification question. This highlights the value of leveraging human feedback to improve model accuracy in classification tasks. Our approach outperforms baselines ranging across all numbers of interactions.

Human Evaluation Table 2 shows the human evaluation results of our full model and two baselines on the FAQ Suggestion and Bird Identification tasks. The model achieves Accuracy@1 of 30% and 20% for FAQ and Bird tasks respectively, when there is no interaction. 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 achieves the best performance: an Accuracy@1 of 59% for FAQ Suggestion and 45% for Bird Identification. Qualitatively, the users rate our full model to be more rational. The human evaluation demonstrates that our model handles real user interaction more effectively despite being trained with only non-interactive data. Appendix A.3 includes additional details for human evaluation and exemple interactions.

Learning behavior Figure 3 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 behavior during the first $1,000$ training episodes in the middle and the right plots. The models achieve relatively stable accuracy after the early exploration stage. As expected, the three runs end up using different numbers of expected turns due to the choice of different r_a values.

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 or multi-choice questions. Our method uses information theory to select the best question

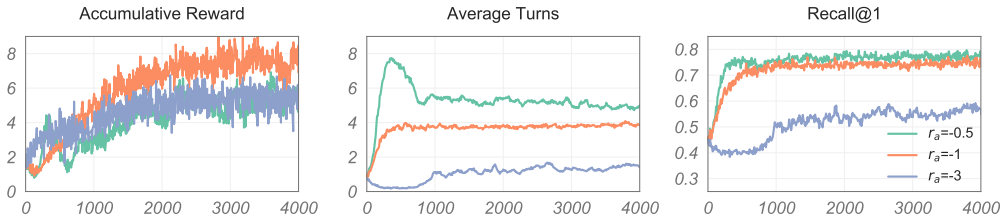

Figure 3: Learning curves of our full model. We show accumulative reward (left), interaction turns (middle), and Accuracy@1 (right) on the test set, where x-axis is the number of episodes run (400 trials per episode). The results are compared on different turn penalty r_a.

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 a large margin. In addition, we provide a new annotated dataset for future work on bootstrapping interactive classification systems.

REFERENCES

Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W. Bruce Croft. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pp. 475–484, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6172-9. doi: 10.1145/3331184.3331265. URL http://doi.acm.org/10.1145/3331184.3331265.

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.

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.

Cen Chen, Chilin Fu, Xu Hu, Xiaolu Zhang, Jun Zhou, Xiaolong Li, and Forrest Sheng Bao. Reinforcement learning for user intent prediction in customer service bots. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pp. 1265–1268, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6172-9. doi: 10.1145/3331184.3331370. URL http://doi.acm.org/10.1145/3331184.3331370.

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.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2017.

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.

Marin Ferecatu and Donald Geman. Interactive search for image categories by mental matching. In *2007 IEEE 11th International Conference on Computer Vision*, pp. 1–8. IEEE, 2007.

Xiaoxiao Guo, Hui Wu, Yu Cheng, Steven Rennie, Gerald Tesauro, and Rogerio Schmidt Feris. Dialog-based interactive image retrieval, 2018.

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.

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.

In-Ho Kang and GilChang Kim. Query type classification for web document retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 64–71. ACM, 2003.

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.

Adriana Kovashka and Kristen Grauman. Attribute pivots for guiding relevance feedback in image search. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 297–304, 2013.

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.

Sang-Woo Lee, Tong Gao, Sohee Yang, Jaejun Yoo, and Jung-Woo Ha. Large-scale answerer in questioner's mind for visual dialog question generation, 2019.

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.

Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention, 2018.

Chen Qu, Liu Yang, W. Bruce Croft, Yongfeng Zhang, Johanne R. Trippas, and Minghui Qiu. User intent prediction in information-seeking conversations. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*, CHIIR '19, pp. 25–33, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6025-8. doi: 10.1145/3295750.3298924. URL http://doi.acm.org/10.1145/3295750.3298924.

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.

Daniel E Rose and Danny Levinson. Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web*, pp. 13–19. ACM, 2004.

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.

Pushkar Shukla, Carlos E. L. Elmadjian, Richika Sharan, Vivek Kulkarni, Matthew Turk, and William Yang Wang. What should I ask? using conversationally informative rewards for goal-oriented visual dialog. *CoRR*, abs/1907.12021, 2019. URL http://arxiv.org/abs/1907.12021.

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.

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.

Ziyu Yao, Yu Su, Huan Sun, and Wen tau Yih. Model-based interactive semantic parsing: A unified framework and a text-to-sql case study, 2019.

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. To make the annotation efficient, we rank the tag-document relevance using the model trained on the previously collected query data. We then take the top 50 possible tags for each document and split them into five non-overlapping lists (i.e. ten tags for each list). Each of the list is assigned to four separate workers to annotate. In the FAQ task, we observe that showing only the top-50 tags out of 813 is sufficient. Figure A.1 illustrates this: after showing the top-50 tags, the curve plateaus and no new tags are assigned to a target. Our annotator agreement is illustrated in A.1 as the Cohen's κ score.

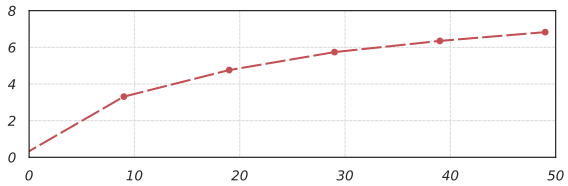

Figure A.1: Accumulated number of tags assigned to the targets (y-axis) by AMT against tag ranking (x-axis). The ranking indicates the relevance of the target-tag pairs from the pre-trained model. The curve plateaued at rank 50 suggests that the lower ranked tags are less likely to be assigned to the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

	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 κ	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 κ is the mean pairwise Cohen's κ score.

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.

We use initial queries as well as paraphrase queries to train the encoder, which has around 16K target-query examples. The breakdown analysis is shown in Table A.2. To 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. On the other hand, when we use the set of tags instead of initial query as text input for a specific target label, the classification performance improves, indicating the designed tags can capture the target label well. Finally, when we concatenate user initial query and tags and use that as text input to the classifier, we achieve Accuracy@1 of 76%. In our full model, we achieve 79% with only querying about 5 tags, indicating effectiveness of our modelling.

Text Input		**Init Query**		**Init Query + Tags**		**Init + Paraphrase Query**		**Full Data**	
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@$\{1, 3\}$.

A.3 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.2. 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.3. Our full model performs best on Accuracy@1, naturalness, and rationality. We show human evaluation examples in Table A.4.

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 binary questions from the same label group. For example, if user replied 'not visible' to a the question 'does the bird has black tail?', then question 'does the bird has yellow tail?', 'does the bird has red tail?' etc. won't be asked again. 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.2. 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 3 turns of interaction. For Bird Identification, the annotators reported that the predicted image is almost identical to the true image sometimes. To better understand the task and the model behavior, we show the confusion matrix of the final model prediction after interaction in Figure A.3. 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.4 show how the confusion matrix change, adding the interactions. The model makes improvement in distinct and also similar bird types.

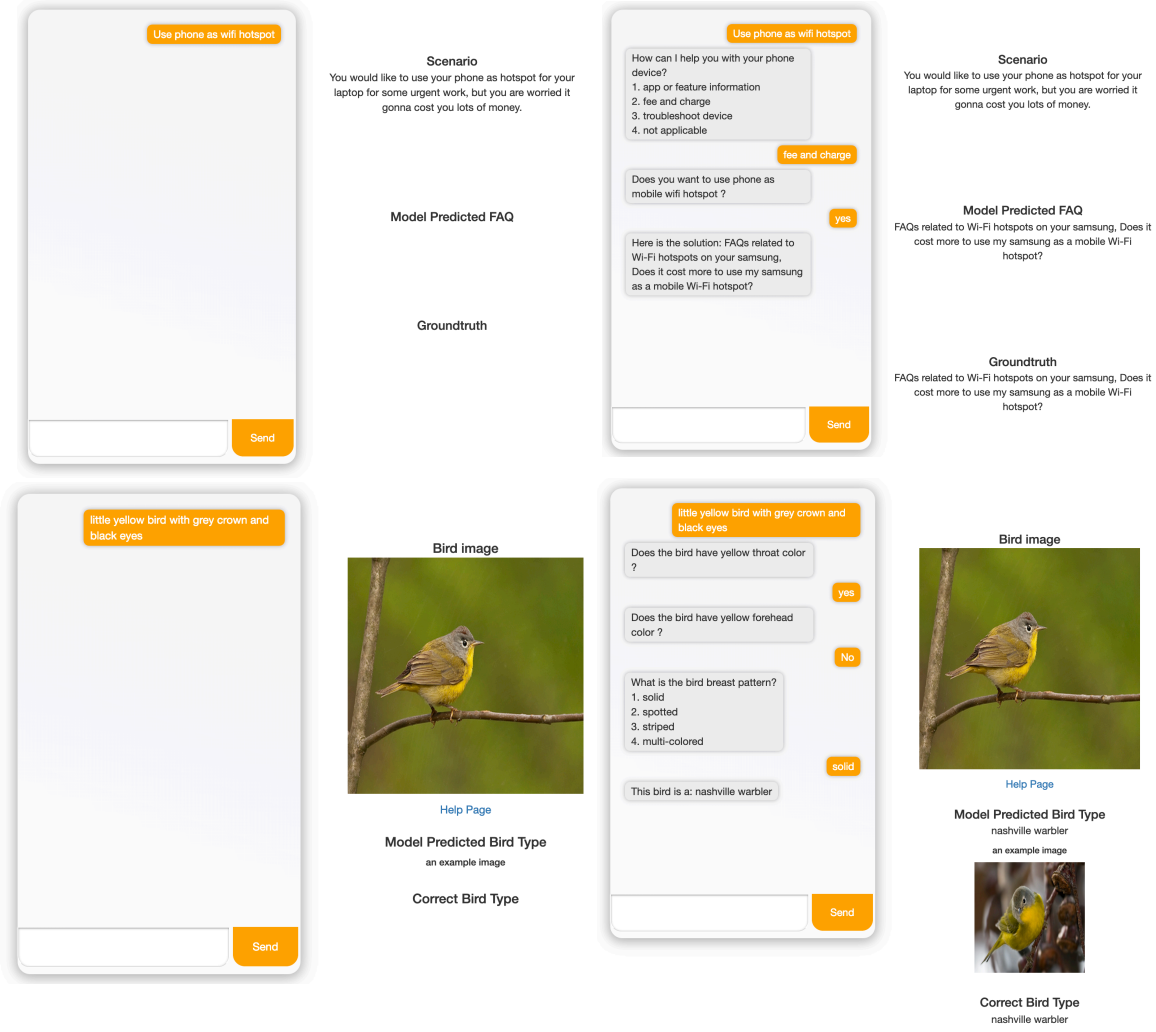

Figure A.2: User interface for FAQ suggestion task (top) and bird identification (bottom) tasks. Left panel shows the interface at the beginning of the interaction and the right panel shows the inter face at the end of the interaction

Dataset	Models	Count	Initial Acc@1	Final Acc@1	Naturalness	Rationality
FAQ	Ours	60	30%	59%	0.4	0.81
	Ours(Fixed Turn)	55	30%	56%	-0.17	0.45
	Static Interact	55	28%	45%	0.03	0.41
Bird	Ours	60	21%	45%	0.21	0.95
	Ours(Fixed Turn)	55	13%	37%	-0.50	0.55
	Static Interact	55	25%	28%	0.1 0	0.85

Table A.3: Human evaluation results on FAQ Suggestion and Bird Identification on our proposed model and several baslines. The three FAQ systems ask 2.8, 3 and 3 turns of questions, respectively. The three Bird 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.

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:	Battery ran out immediately.
S:	How can I help you with your phone device?
	App or feature information; Fee and charge; Troubleshooting; Not applicable
U:	Troubleshooting.
S:	Are you using mobile wifi hotspot?
U:	Yes.
S:	Please read: *Does using my phone as a mobile Wi-Fi hotspot drain my battery?*

Table A.4: Examples of user interactions for FAQ suggestion human evaluation.

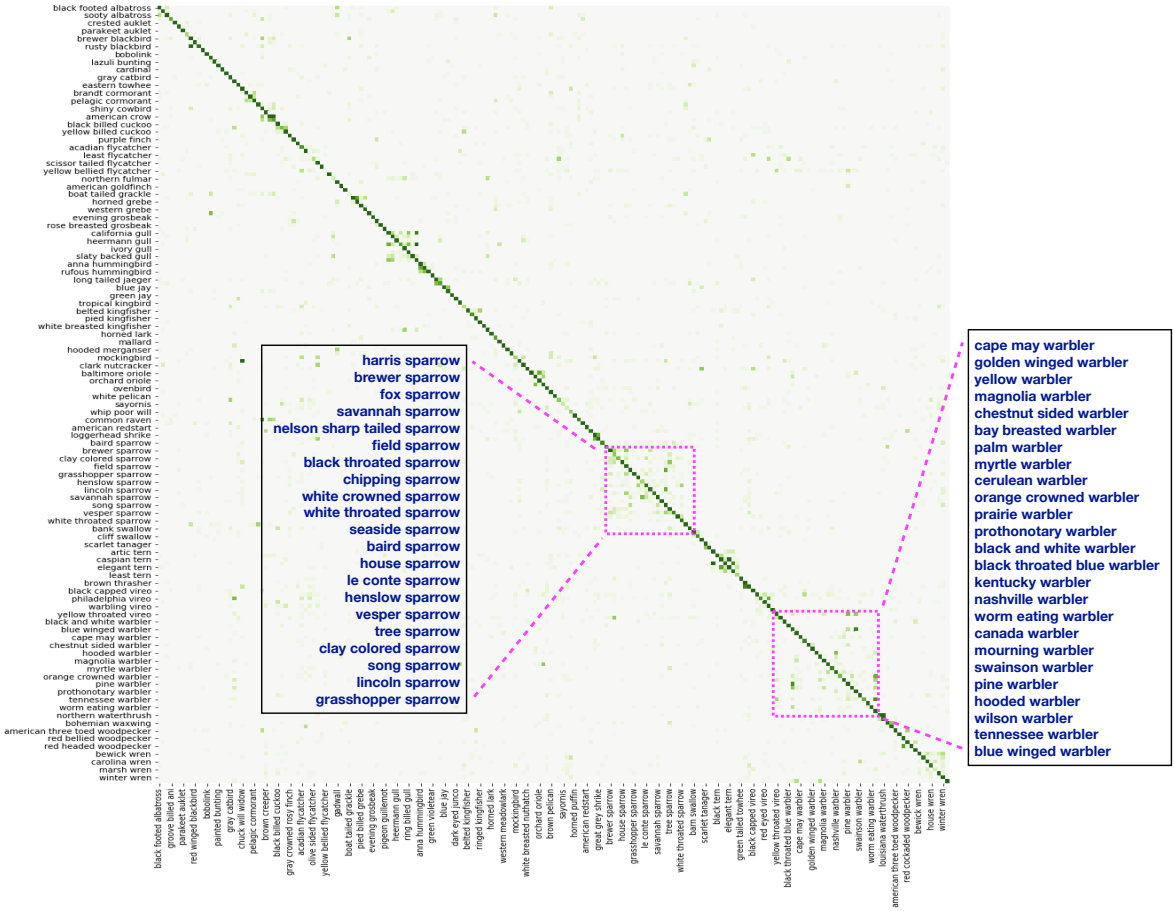

Figure A.3: Confusion matrix of our final output for bird identification task.

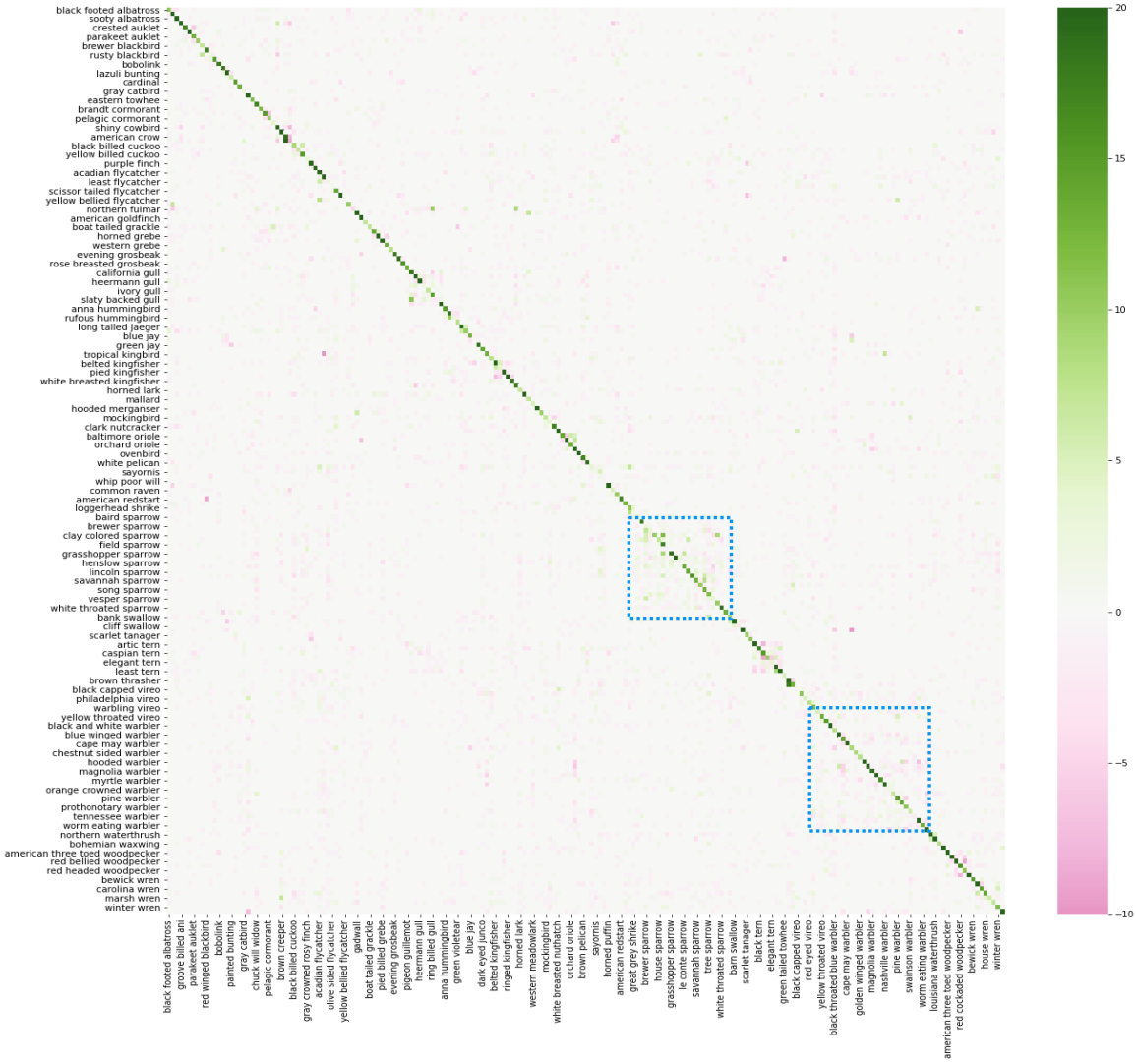

Figure A.4: Confusion matrix difference between the initial query with and without the interactions. We desire high value in the diagonal part and low value elsewhere.