Dialog Policy Learning for Joint Clarification and Active Learning Queries

Aishwarya Padmakumar
Department of Computer Science
University of Texas at Austin
aish@cs.utexas.edu

Raymond Mooney
Department of Computer Science
University of Texas at Austin
mooney@cs.utexas.edu

Abstract

Intelligent systems need to be able to recover from mistakes, resolve uncertainty, and adapt to novel concepts not seen during training. Dialog interaction can enable this by the use of clarifications for correction and resolving uncertainty, and active learning queries to learn new concepts encountered during operation. Prior work on dialog systems has either focused on exclusively learning how to perform clarification/information seeking, or to perform active learning. In this work, we train a hierarchical dialog policy to jointly perform both clarification and active learning in the context of an interactive language-based image retrieval task motivated by an on-line shopping application, and demonstrate that jointly learning dialog policies for clarification and active learning is more effective than the use of static dialog policies for one or both of these functions.

1 Introduction

The ability to understand and communicate in natural language can improve the accessibility of systems such as robots, home devices and computers to non-expert users. Voice assistant applications that understand high level instructions in natural language are increasingly becoming a part of a variety of devices. Since language can often be ambiguous, it would be desirable for such systems to engage in a dialog with the user to clarify their intentions and obtain missing information.

The specific environment in which a system is deployed may also contain domain specific vocabulary or concepts that were not encountered during training. For example, a dialog system in a shopping domain may need to be updated with the introduction of new clothing styles. Hence it is desirable for a system to adapt its models to the operating environment using information from user interactions.

| Speaker   | Action Type         | Simulated Dialog Act                      | Natural Language Gloss                   |
|-----------|---------------------|-------------------------------------------|------------------------------------------|
| Agent     | -                   | Is there something I can help you find?   |
| User      | Target Description  | Mixed Pain Signals Spaghetti Top          | I’m looking for a mixed pain signals spaghetti top |
| Agent     | Clarification       | Clarification: Pink                       | Would you like one which is pink?        |
| User      | Binary Label        | 1                                         | Yes                                      |
| Agent     | Label Query         | Label Query: Sleeveless                   | Would you describe this item as sleeveless? |
| User      | Binary Label        | 1                                         | Yes                                      |
| Agent     | Example Query       | Example Query: Spaghetti                  | Can you share an image of an item with spaghetti straps? |
| User      | Image               |                                            | Here is an example,                      |
| Agent     | Guess               |                                            | Is this what you were searching for?     |
| User      | Success: O'1        | 1                                         | Yes                                      |

Figure 1: A stylized sample interaction. In this work, we use simulated dialog acts but the natural language glosses represent how such a dialog could look in an end application.
Prior work on dialog and user interaction typically focuses either exclusively on clarification/information-seeking settings \cite{36,23}, or building/improving models through active learning \cite{35,22}. In this work, we train a hierarchical dialog policy to jointly learn to choose clarification and active learning queries in an interactive image retrieval task in the fashion domain. A sample interaction is shown in Figure 1. We consider an application where a dialog system is combined with a retrieval system to help a customer find an article of clothing. Instead of showing a large number of results obtained from the retrieval system, the dialog system attempts to use clarifications to refine the search query, and active learning questions to obtain labelled examples for concepts it has not been trained on.

Most task-oriented dialog tasks require the system to identify one or more goals specified by a user, using a slot-filling model \cite{36}. These can be considered as learning to choose between a set of clarification questions that can confirm or obtain the value of various slots. However, for tasks such as natural language image retrieval, it is non-trivial to extend the slot-filling paradigm to perform clarification, as there is no standard set of slots that natural language descriptions of images can be divided into. Also, learned models are needed to identify components such as objects or attributes, and it is difficult to enumerate all expected types of these.

Some tasks such as GuessWhat?! \cite{6} or discriminative question generation \cite{18} allow the system to ask unconstrained natural language clarification questions. However in these settings, specially designed models are still needed to ensure that learned questions actually decrease the size of the search space \cite{17,38}. Such open ended questions are also difficult to answer in simulation, which is often necessary for learning good dialog policies. Hence, in these tasks, the system often learns to ask “easy” questions that can be reliably answered by a learned answering module \cite{39}.

In this work, we explore a middle-ground approach with a form of attribute-based clarification \cite{11}. We use the term “attribute” to refer to a mix of concepts including categories such as “shirt” or “dress”, more conventional attributes such as colors, and domain specific attributes such as “sleeveless” and “V-neck”. Although we work with a dataset that contains a fixed set of attributes annotated for each image, we simulate the setting where novel visual attributes are encountered at test time.

Dialog interaction can also be used to improve an underlying model using opportunistic active learning \cite{22}. Active learning allows a system to identify unlabeled examples that, if labeled, are most likely to improve the underlying model. Opportunistic active learning \cite{30} incorporates such queries into an interactive task in which an agent may ask ordinary users questions that are irrelevant to the current dialog interaction to improve performance in future dialog interactions. Opportunistic queries are more expensive than traditional active learning queries as they may distract from the task at hand, but they can allow the system to perform more effective lifelong learning. Such queries have been shown to improve performance in interactive object retrieval \cite{22}. However, this, and other works in learning reinforcement learning (RL) policies for active learning \cite{10} do not account for the presence of other interactive actions such as clarification.

In this work, we set up a dialog task that combines natural language image retrieval with both opportunistic active learning and attribute-based clarification. We then learn hierarchical dialog policies that are jointly learned to choose both appropriate clarification and active learning questions in a setting containing both uncertain visual classifiers and novel concepts not seen during training. We observe that in our challenging setup, it is necessary to jointly learn dialog policies for choosing clarification and active learning questions to improve performance over employing one-shot retrieval with no interaction.

2 Related Work

Slot-filling style clarifications \cite{36} have been shown to be useful for a variety of domains including restaurant recommendation \cite{34}, restaurant reservation \cite{4}, movie recommendation and question answering \cite{9}, issuing commands to robots \cite{7,31} and converting natural language instructions to code \cite{5}. Other tasks such as GuessWhat?! \cite{6}, playing 20 questions \cite{15}, relative captioning \cite{14} and discriminative question generation \cite{18} enable very open-ended clarification. In this work, we choose an intermediate class of clarification questions that allow finer-grained clarification than the slot filling model, but are still somewhat constrained so that reasonably correct answers can be provided in simulation.
Most of the above works learn dialog policies for clarification using RL \([23, 33, 28, 15]\). Some of the above systems use information from clarifications to improve the underlying language-understanding model \([21, 23]\). Such improvement is implicit in end-to-end dialog systems \([33, 6, 15]\). However, we use explicit active learning questions to improve the underlying perceptual model used for language grounding. There exist some previous works that use visual attributes for clarification \([8, 24]\) but these do not use this information for improving the underlying language understanding model.

Active learning has traditionally been done using various hand-coded sample-selection metrics, such as uncertainty sampling \([27]\). Recent work on using active learning in dialog/RL setups include using slot-filling style active learning questions to learn new names for known perceptual concepts \([17]\), sequentially examining examples and deciding whether or not to query for their labels, up to a budget \([10]\), deciding between predicting a label for a specific example or requesting for it to be labelled \([35]\), and jointly learning a data selection heuristic, data representation, and prediction function \([1]\). However, most of these (except \([35]\)) do not involve a trade-off between active learning and task completion. None of them incorporate clarification questions.

Most similar to our work is \([23]\) which concerns learning a policy to trade-off opportunistic active learning questions to improve classifiers, and using these to ground natural-language descriptions of objects. However, instead of considering the cold-start condition where the system cannot initially ground any descriptions before some questions are asked, we consider a warm-start condition which is closer to most real-world scenarios. We use a pretrained classifier and expect active learning to primarily aid generalization to novel concepts not seen during training. We also extend the task to include clarification questions.

Also related is work on interactive image retrieval such as allowing a user to mark relevant and irrelevant results \([21, 32]\), which acts as a form of clarification. Recent works allow users to provide additional feedback using language to refine search results \([14, 3, 26]\). These directions are complementary to our work and can potentially be combined with it in the future.

## 3 Task Setup

We consider an interactive task of retrieving an image of a product based on a natural-language description. Given a set of candidate images and a natural-language description, the goal of the system is to identify the image being referred to. Before trying to identify the target image, the system can ask the user a combination of both clarification and active learning questions. The goal of the task is to maximize the number of correct product identifications across interactions, while also keeping dialogs as short as possible.

Since we want to ensure that active learning questions are used to learn a generalizable classifier, we follow the setup of \([22]\) and in each interaction we present the system with two sets of images:

- **An active test set** \(I^{Te}\) consisting of the candidate images that the description could refer to.
- **An active training set** \(I^{Tr}\) which is the set of images that can be queried for active learning.

It is also presented with a description of the target image. Before attempting to identify the target, the system can ask clarification or active learning questions. We assume the system has access to a set of attributes \(W\) that can be used in natural language descriptions of products. Given these attributes, the types of questions the system can ask are as follows (see Figure 1 for examples of each):

- **Clarification query** - A yes/no query about whether an attribute \(w \in W\) is applicable to the target.
- **Label query** - A yes/no query about whether an attribute \(w \in W\) is applicable to a specific image \(i\) in the active training set \(I^{Tr}\).
- **Example query** - Ask for a positive example in the active training set \(I^{Tr}\) for an attribute \(w \in W\).

The dialog ends when the system makes a guess about the identity of the target and is considered successful if it this is correct. The goal of the task is to maximize the number of successful guesses, while also keeping dialogs as short as possible. As in \([22]\), we allow label and example queries that are either on-topic – queries about attributes in the current description– or opportunistic – queries that are not relevant to the current description but may be useful for future interactions, which have been shown to be helpful for interactive object retrieval \([30]\) (see Figure 1 for examples of each).
4 Methodology

4.1 Visual Attribute Classifier

We train a multilabel classifier for predicting visual attributes given an image. The network structure for the classifier is shown in Figure 2. We extract features for the images using the penultimate layer of an Inception-V3 network [29] pretrained on ImageNet [25]. These are passed through two separate fully connected (FC) layers with ReLU activations, that are summed to produce the final layer used for classification. This is converted into per-class probabilities \( p(i) \) using a sigmoid layer with temperature correction [12]. We obtain another set of per-class probabilities \( p'(i) \) by passing the output of one of the FC layers through a sigmoid layer with temperature correction. Mathematically, given features \( \phi(i) \) for image \( i \), we have,

\[
\begin{align*}
    f(i) &= \text{ReLU}(w^T \phi(i) + b) \\
    p(i) &= \sigma(f(i) \odot \frac{1}{\tau}) \\
    f'(i) &= \text{ReLU}(w'^T \phi(i) + b') \\
    p'(i) &= \sigma(f'(i) \odot \frac{1}{\tau'})
\end{align*}
\]

where \( w, w', \tau \) and \( \tau' \) are learned vectors and \( b \) and \( b' \) are learned biases.

We train the network using a loss function that combines cross-entropy loss on \( p(i) \) over all examples with the cross entropy loss over \( p'(i) \) only for positive labels. That is,

\[
L = (1 - \lambda) \sum_i y_i \log p(i) + (1 - y_i) \log(1 - p(i)) + \lambda \sum_i y_i \log p'(i)
\]

where \( y_i \) is the label vector for image \( i \). This forces part of the network to focus on positive examples for each class. This is required because we use a heavily imbalanced dataset where most classes have very few positive examples. We find this more effective than a standard weighted cross entropy loss, and the results in this paper use \( \lambda = 0.9 \). We also maintain a validation set of images labeled with attributes, that can be extended using active learning queries. Using this, we can estimate per-attribute precision, recall and F1. These metrics are used for tuning classifier hyperparameters and for dialog policy learning.

4.2 Grounding Model

We assume that a description is a conjunction of attributes, and use heuristics based on string matching to determine the set of attributes referenced by the natural language description. Let the subset of attributes referenced in the description be \( W_d \subseteq W \).

Suppose we additionally obtain from clarifications that attributes \( W_p \subseteq W \) apply to the target image, and attributes \( W_n \subseteq W \) do not apply, assuming independence of attributes, the probability that \( i \) is the target image, \( b(i) \) is:

\[
b(i) \propto \prod_{w \in W_d} p_w(i) \prod_{w \in W_p} p_w(i) \prod_{w \in W_n} (1 - p_w(i))
\]

At any stage, the best guess the system can make is the image with max belief, that is

\[i_{\text{guess}} = \arg\max_{i \in I} b(i)\]

Also, we estimate the information gain \( J \) of a clarification \( q \in W \) as follows. This is based on the formulation used in [17] but we additionally make a Markov assumption.

\[
J(q) = \sum_{i \in I} \sum_{a \in \{0,1\}} b(i) P(a|q, i) \ln \left( \frac{P(a|q, i)}{P(a|q)} \right)
\]
where $P(1|q,i) = p_q(i)$ and $P(0|q,i) = 1 - p_q(i)$ and $P(a|q) = \sum_i b(i)P(a|q,i)$.

### 4.3 MDP Formulation

We model each interaction as an episode in a Markov Decision Process (MDP) where the state consists of the images in the active training and test sets, the attributes mentioned in the target description, the current parameters of the classifier, and the set of queries asked and their responses. At each state, the agent has the following available actions:

- A special action for guessing – the image is chosen using Equation 2
- One clarification query per attribute.
- A set of actions corresponding to possible active learning queries – one example query per attribute and one label query corresponding to each pair $(w,i)$ for $w \in W$, $i \in I^{Tr}$.

We do not allow actions to be repeated. We learn a hierarchical dialog policy composed of 3 parts – clarification and active learning policies to respectively choose the best clarification and active learning query in the current state, and a decision policy to choose between clarification, active learning, and guessing. An episode ends either when the guess action is chosen, or when a dialog length limit is reached, at which point the system is forced to make a guess. If the episode ends with a correct guess, the agent gets a large positive reward. Otherwise the agent gets a large negative reward at the end of the episode. Additionally, the agent gets a small negative reward for each query to encourage shorter dialogs. In our experiments, we treat these rewards as tunable hyperparameters.

### 4.4 Policy Learning

We experimented with using both Q-learning and A3C [20] for policy learning, both trained to maximize the discounted reward. Since the classifier has a large number of parameters, it is necessary to extract task-relevant features to represent state-action pairs. The features provided to each policy need to capture information from the current state that enable the system to identify useful clarifications and active learning queries, and trade off between these and guessing. The features used include:

### 4.5 Clarification policy features

- Metrics about the current beliefs $\{b(i) : i \in I^{Te}\}$ and what they would be for each possible answer, if the question were asked:
  - Entropy: A higher entropy suggests that the agent is more uncertain. A decrease in entropy could indicate a good clarification.
  - Top two highest beliefs and their difference: A high value of the maximum belief, or a high difference between the top two beliefs could indicate that the agent is more confident about its guess. An increase in these could indicate a good clarification.
  - Difference between the maximum and average beliefs: A large difference suggests that the agent is more confident about its guess. An increase in these could indicate a good clarification.
- Information gain of the query as calculated in section 4.2.
- Current F1 of the attribute associated with the query: The system is likely to make better clarifications using attributes with high predictive accuracy.

### 4.6 Active learning policy features

- Current F1 of the attribute associated with the query, since the system is likely to benefit more from improving an attribute whose current predictive accuracy is not high.
- Fraction of previous dialogs in which the attribute has been used, since it is beneficial to focus on frequently used attributes that will likely benefit future dialogs.
- Fraction of previous dialogs using the attribute that have been successful, since this suggests that the attribute may be modelled well enough already.
- Whether the query is off-topic (i.e. opportunistic), since this would not benefit the current dialog.

Additionally in label queries,
• For query \((w, i)\), \(|p_w(i) - 0.5|\) as a measure of (un)certainty.
• Average cosine distance of the image to others in the dataset; this is motivated by density weighting to avoid selecting outliers.
• Fraction of k-nearest neighbors of the image that are unlabelled for this attribute, since a higher value suggests that the query could benefit multiple images.

4.7 Decision policy features

• Features of the current belief as in section 4.5. These can be used to determine whether a guess is likely to be successful.
• Information gain of the best clarification action – to decide the utility of the clarification.
• Margin from the best active learning query if it is a label query – to decide the utility of the label query.
• F1 of attributes in clarification and active learning queries. High F1 is desirable for clarification and low F1 for active learning.
• Mean F1 of attributes in the description. A high value suggests that the belief is more reliable.
• Number of dialog turns completed.

4.8 Baseline static policy

As a baseline, we use an intuitive manually-designed static policy that is also hierarchical and was tailored to perform well in preliminary experiments. The static clarification policy chooses the query (among those with F1 > 0) with maximum information gain. Ties are broken using F1 of the attribute in the query. The static active learning policy has a fixed probability of choosing label queries and example queries. Uncertainty sampling is used to select the label query \((w, i)\) with minimum \(|p_w(i) - 0.5|\). An example query is chosen uniformly at random from the candidates. The decision policy initially chooses clarification if the information gain is above a minimum threshold, and the highest belief is below a confidence threshold. After a maximum number of clarifications, it chooses active learning until another threshold on the dialog length before guessing.

5 Experimental Setup

5.1 Dataset

To address a potential shopping application, we simulate dialogs using the iMaterialist Fashion Attribute dataset [13], consisting of images from the shopping website Wish each annotated for a set of 228 attributes. We scraped product descriptions for the images in the train and validation splits of the dataset for which attribute annotations are publicly available. After removing products whose images or descriptions were unavailable, we had 648,288 images with associated product descriptions and annotations for the 228 attributes.

We create a new data split following the protocol of [22] to ensure that the learned dialog policy generalizes to attributes not seen during policy training. We divided the dataset into 4 splits, \(policy_{pretrain}\), \(policy_{train}\), \(policy_{val}\) and \(policy_{test}\), such that each contains images that have attributes for which positive examples are not present in earlier splits. More details are included in Appendix B. Each of these is then split into subsets \(classifier_{training}\) and \(classifier_{test}\) by a uniform 60-40 split.

The \(policy_{pretrain}\) data is used to pretrain the multi-class attribute classifier. We use its \(classifier_{training}\) subset of images for training and its \(classifier_{test}\) subset to tune hyperparameters. The \(policy_{train}\) data is then used to learn the dialog policy. The \(policy_{val}\) data is used to tune hyperparameters as well as choose between RL algorithms. Results are reported for \(policy_{test}\) data.

We wish to simulate dialogs as refinements of an initial retrieval based on the product description. For the description of each image in the current \(classifier_{test}\) subset, we rank all other images in this subset according to a simplified version of the score in equation 1 (details in Appendix C). From the images which get ranked within the top 1000 for their corresponding description, we sample

\[https://www.wish.com/\]
target images for each interaction. The active test set for the interaction consists of the top 1000 images as ranked for that description. We randomly sample 1000 images from the appropriate classifier_training subset to form the active training set.

In each interaction, the description of the target image is provided to the agent to start the interaction. The annotated attributes are used to answer queries from the system. This simulation procedure is similar to that of [22] but the answers to questions are less noisy in our simulation as all attributes are annotated as positive or negative for all images.

5.2 Experiment Phases

We run dialogs in batches of 100 and update the classifier and policies at the end of each batch. This is followed by repeating the retrieval step for all descriptions in the classifier_test subset before choosing target images for the next batch of dialogs. The experiment has the following phases:

- Classifier pretraining: We pretrain the classifier using annotated attribute labels for images in the classifier_training subset of the policy_pretrain set. This ensures that we have some reasonable clarifications at the start of dialog policy learning.
- Policy initialization: We initialize the dialog policy using experience collected using the baseline static policies (section 4.8) for the decision and active learning policies, and an oracle \(^2\) to choose clarifications. This is done to speed up policy learning. The dialogs for this phase are sampled from the set of policy_train images.
- Policy training: This phase consists of training the policy using on-policy experience, with dialogs again sampled from the set of policy_train images.
- Policy testing: We reset the classifier to the state at the end of pretraining. This is done to ensure that any performance improvement seen during testing are due to queries made in the testing phase. This is needed both for fair comparison with the baseline and to confirm that the system can generalize to novel attributes not seen during any stage of training. Dialogs are sampled for this from the policy_val set for hyperparameter tuning and from the policy_test set for reported results.

6 Results and Discussion

We initialize the policy with 4 batches of dialogs, followed by 4 batches of dialogs for the training phase, and 5 batches of dialogs in the testing phase. We compare the fully learned policy with hierarchical policies that consist of keeping one or more of the components static. We also compare the choice of Q-Learning or A3C [20] as the policy learning algorithm for each learned policy. Table 1 shows the performance in the final test batch of the best fully learned policy, as well as a selected subset of the baselines (all conditions are included in Appendix A). We evaluate policies on the fraction of successful episodes in the final test batch, and the average dialog length.

| Decision Type | Policy Type | Clarification Policy Type | Active Learning Policy Type | Fraction of Successful Dialogs | Average Dialog Length |
|---------------|-------------|--------------------------|----------------------------|-------------------------------|-----------------------|
| Q-Learning    | A3C         | A3C                      | 0.33                       | 9.40                          |
| Q-Learning    | A3C         | Static                   | 0.15                       | 14.16                         |
| Q-Learning    | Static      | A3C                      | 0.09                       | 1.00                          |
| Static        | A3C         | A3C                      | 0.27                       | 20.00                         |
| Static        | Static      | Static                   | 0.17                       | 20.00                         |

Ideally we would like the system to have a high dialog success rate while having as low a dialog length as possible. We observe that the using a learned policy for all three functions results in a significantly more successful dialog system (according to an unpaired Welch t-test with \(p < 0.05\)) than most conditions in which one or more of the policies are static. The exception is the case when

---

\(^2\)The oracle tries each candidate clarification and returns the one that maximally increases the belief of the target image.
the decision policy is static and the clarification and active learning policies are learned, in which case the difference is not statistically significant. The fully learned policy also uses significantly shorter dialogs than all conditions with a static decision policy. Some other conditions result in shorter dialogs, but these are unable to exploit the clarification and active learning actions enough to result in a success rate comparable to the fully learned policy.

Figure 3 plots the success rate across test batches, and the expected success rate if the system was forced to guess without clarification, for the fully learned, and fully static policies. We find that in the case of the fully static policy, there is no statistically significant improvement, either in the expected initial success rate without clarifications, or in the final success rate, between the first and last test batch. This suggests that neither the static active learning policy, nor its combination with the static clarification policy are capable of improving the system’s performance.

However, in the case of the fully learned policy, we observe a statistically significant improvement in the final success rate, but not the initial success rate without clarifications. This suggests that while a learned active learning policy by itself is not sufficient to improve the system’s success rate, the combination of learned active learning and clarification policies is sufficient to improve the system’s success rate. We also observe that while the difference between the initial and final success rate is initially not significant, it increases across batches, and becomes significant in the last two batches. This suggests that the clarification policy by itself is also insufficient for improvement, and the combination of the two is required to improve the system’s success rate.

Figure 3: Comparison of guess success rate with, and without clarifications across test batches

We believe that the reason for the relatively poor performance of the static clarification and active learning policies is that the classifier is not sufficiently accurate, and does not produce well calibrated probabilities, due to the heavy imbalance in the dataset. On the other hand, the learned policies are able to learn to properly adjust for this miscalibration. We also believe that with more training dialogs or a different state-action representation, it may be possible to also learn a decision policy that outperforms the static decision policy.

7 Conclusion

We demonstrate how a combination of RL learned policies for choosing attribute-based clarification and active learning queries can be used to improve an interactive system that needs to retrieve images based on a natural language description, while encountering novel attributes at test time not seen during training. Our experiments show that in challenging datasets where it is difficult to obtain an accurate attribute classifier, learned policies for choosing clarification and active learning queries outperform strong static baselines. We further show that in this challenging setup, a combination of learned clarification and active learning policies is necessary to obtain improvement over directly performing retrieval without interaction.
**Broader Impact**

Natural language interfaces such as language-based search, and intelligent personal assistants have the potential to make various forms of technology ranging from mobile phones and computers, as well as robots or other machines such as ATMs or self-checkout counters more accessible and less intimidating to users who are unfamiliar or uncomfortable with other interfaces on such devices such as command shells, button based interfaces or changing visual user interfaces. Spoken language interfaces can also be used to make such devices more accessible for the visually impaired or users who have difficulty with fine motor control.

However, the use of these interfaces do involve concerns over privacy and data security. This is especially the case with devices based on spoken language interfaces as they need to analyze every conversation for potential codewords. Thus, users need to trust that these extraneous conversations will not be stored, or analyzed for other information. This is particularly problematic in environments such as hospitals or lawyer’s officers where confidentiality is expected.

Another concern is that transactions on these devices may be triggered by casual conversation or voices on television, that were not intended to activate the dialog system. A related concern is that the ambiguity of language or mistakes made by the system may trigger unintended actions. In most applications, these can be handled by setting up appropriate confirmation or cancellation procedures for sensitive actions. Increased use of clarification steps before execution of an action may provide an additional opportunity for users to cancel such actions before they take place.

Using active learning, or any form of continuous learning with user data can make machine learning systems more useful due to increased exposure to the data distribution that such systems need to operate on in practice. However, most machine learning algorithms assume that the input data is complete and correct, both of which may be violated by systems that train on user-generated data. It is also possible for such data to be biased in a variety of ways – ranging from potential absence of representation or misrepresentation of some groups of people who do not use the system as frequently, to filter-bubble like effects when many users provide a few frequent examples as training data to the system. Explicit active learning questions also allow users to deliberately provide misinformation to machine learning systems. Practical systems using active learning need to incorporate methods for handling noisy data, and need to have tests in place for undesirable learned biases.

**Acknowledgments and Disclosure of Funding**

We would like to thank Prasoon Goyal, Peter Stone, Joydeep Biswas and the UT Austin Building Wide Intelligence group for helpful discussions.

This work was funded by a Google Faculty Research Award (2019-2020) that was awarded to Prof Ray Mooney, an NSF NRI grant (IIS-1637736): Robots that Learn to Communicate through Natural Human Dialog, and an NSF NRI 2.0 grant (IIS-1925082): Improving Robot Learning from Feedback and Demonstration using Natural Language.

**References**

[1] Philip Bachman, Alessandro Sordoni, and Adam Trischler. Learning algorithms for active learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 301–310. JMLR. org, 2017.

[2] Ricardo Baeza-Yates. Data and algorithmic bias in the web. In *Proceedings of the 8th ACM Conference on Web Science*, pages 1–1, 2016.

[3] Indrani Bhattacharya, Arkabandhu Chowdhury, and Vikas C Raykar. Multimodal dialog for browsing large visual catalogs using exploration-exploitation paradigm in a joint embedding space. In *Proceedings of the 2019 on International Conference on Multimedia Retrieval*, pages 187–191, 2019.

[4] Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

[5] Shohbit Chaurasia and Raymond J. Mooney. Dialog for language to code. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP-17)*, pages 175–180, Taipei, Taiwan, November 2017.
[6] Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. Guesswhat?! visual object discovery through multi-modal dialogue. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[7] Robin Deits, Stefanie Tellex, Pratiksha Thaker, Dimitar Simeonov, Thomas Kollar, and Nicholas Roy. Clarifying commands with information-theoretic human-robot dialog. J. Hum.-Robot Interact., 2(2):58–79, June 2013.

[8] Haris Dindo and Daniele Zambuto. A probabilistic approach to learning a visually grounded language model through human-robot interaction. In 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 790–796. IEEE, 2010.

[9] Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. Evaluating prerequisite qualities for learning end-to-end dialog systems. In 4th International Conference on Learning Representations, ICLR 2016, 2016.

[10] Meng Fang, Yuan Li, and Trevor Cohn. Learning how to active learn: A deep reinforcement learning approach. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. ACL, 2017.

[11] Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 1778–1785. IEEE, 2009.

[12] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning - Volume 70, ICML'17, page 1321–1330. JMLR.org, 2017.

[13] Sheng Guo, Weilin Huang, Xiao Zhang, Prasanna Srikanta, Yin Cui, Yuan Li, Hartwig Adam, Matthew R Scott, and Serge Belongie. The imaterialist fashion attribute dataset. In Proceedings of the IEEE International Conference on Computer Vision Workshops, pages 0–0, 2019.

[14] Xiaoxiao Guo, Hui Wu, Yu Cheng, Steven Rennie, Gerald Tesauro, and Rogerio Feris. Dialog-based interactive image retrieval. In Advances in Neural Information Processing Systems, pages 678–688, 2018.

[15] Huang Hu, Xianchao Wu, Bingfeng Luo, Chongyang Tao, Can Xu, Wei Wu, and Zhan Chen. Playing 20 question game with policy-based reinforcement learning. In EMNLP, 2018.

[16] Richard Lackes, Markus Siepermann, and Georg Vetter. Can i help you?–the acceptance of intelligent personal assistants. In International Conference on Business Informatics Research, pages 204–218. Springer, 2019.

[17] Sang-Woo Lee, Yu-Jung Heo, and Byoung-Tak Zhang. Answerer in questioner’s mind for goal-oriented visual dialogue. In Visually-Grounded Interaction and Language Workshop (NIPS), 2018.

[18] Yining Li, Chen Huang, Xiaou Tang, and Chen Change Loy. Learning to disambiguate by asking discriminative questions. In Proceedings of the IEEE International Conference on Computer Vision, pages 3419–3428, 2017.

[19] Andrew Liptak. Amazon’s alexa started ordering people dollhouses after hearing its name on tv, 2017.

[20] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In International conference on machine learning, pages 1928–1937, 2016.

[21] Chahab Nastar, Matthias Mitschke, and Christophe Meilhac. Efficient query refinement for image retrieval. In Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No. 98CB36231), pages 547–552. IEEE, 1998.

[22] Aishwarya Padmakumar, Peter Stone, and Raymond J. Mooney. Learning a policy for opportunistic active learning. In In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-18), Brussels, Belgium, November 2018.

[23] Aishwarya Padmakumar, Jesse Thomason, and Raymond J. Mooney. Integrated learning of dialog strategies and semantic parsing. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017), pages 547–557, Valencia, Spain, April 2017.
A Complete Results

Here, we include the complete set of results of which Table 1 is a part. For each of the three tasks−choosing clarification questions, choosing active learning questions and deciding between these and guessing, we compare the use of a static policy with policies learned using Q-Learning and A3C. We evaluate policies on the fraction of successful episodes in the final test batch, and the average dialog length. An episode is considered successful if it ends with the system guessing the correct
Table 2: Unabridged results from the final batch of the test phase. * indicates the conditions whose performance is comparable to the best condition (in bold).

| Decision Type Policy Type | Clarification Policy Type | Active Learning Policy Type | Fraction of Successful Dialogs | Average Dialog Length |
|---------------------------|---------------------------|----------------------------|-------------------------------|-----------------------|
| Q-Learning Q-Learning    | Q-Learning                | 0.12                       | 15.45                         |
| Q-Learning Q-Learning    | A3C                       | 0.18                       | 16.96                         |
| Q-Learning Q-Learning    | Static                    | 0.14                       | 11.83                         |
| Q-Learning A3C           | Q-Learning                | 0.06                       | 1.0                           |
| Q-Learning A3C           | A3C                       | 0.33                       | 9.4                           |
| Q-Learning A3C           | Static                    | 0.15                       | 14.16                         |
| Q-Learning Static        | Q-Learning                | 0.17                       | 13.96                         |
| Q-Learning Static        | A3C                       | 0.09                       | 1.0                           |
| Q-Learning Static        | Static                    | 0.17                       | 3.81                          |
| A3C Q-Learning           | Q-Learning                | 0.09                       | 20.0                          |
| A3C Q-Learning           | A3C                       | 0.19                       | 20.0                          |
| A3C Q-Learning           | Static                    | 0.17                       | 20.0                          |
| A3C A3C                  | Q-Learning                | 0.13                       | 20.0                          |
| A3C A3C                  | A3C                       | 0.09                       | 20.0                          |
| A3C A3C                  | Static                    | 0.15                       | 20.0                          |
| A3C Static               | Q-Learning                | 0.12                       | 20.0                          |
| A3C Static               | A3C                       | 0.13                       | 20.0                          |
| A3C Static               | Static                    | 0.16                       | 20.0                          |
| Static Q-Learning        | Q-Learning                | *0.29                      | 20.0                          |
| Static Q-Learning        | A3C                       | *0.24                      | 20.0                          |
| Static Q-Learning        | Static                    | 0.1                       | 20.0                          |
| Static A3C               | Q-Learning                | *0.24                      | 20.0                          |
| Static A3C               | A3C                       | *0.27                      | 20.0                          |
| Static A3C               | Static                    | 0.14                       | 20.0                          |
| Static Static            | Q-Learning                | 0.15                       | 20.0                          |
| Static Static            | A3C                       | 0.16                       | 20.0                          |
| Static Static            | Static                    | 0.17                       | 20.0                          |

target item. Ideally, we would like the system to have a high dialog success rate while having as low a dialog length as possible.

We observe that the using a learned policy for all three functions, with A3C for choosing clarification and active learning queries, and Q-learning for deciding between clarification, active learning and guessing, results in a significantly more successful dialog system (according to an unpaired Welch t-test with p < 0.05) than most other conditions. The exceptions are when the decision policy is static and the clarification and active learning policies are learned (marked by * in Table 2), in which the difference is not statistically significant. However, the best fully learned policy uses significantly shorter dialogs than the other policies with a comparable success rate, thus making it overall more desirable.

**B Data Split**

We divide the set of attributes into 4 subsets – policy_pretrain, policy_train, policy_val and policy_test. Using these, we divide the images into 4 subsets as follows:

- All images having a positive label for any of the attributes in policy_test subset form the policy_test set of images.
- Of the remaining images, the images with a positive label for any attribute in the policy_val form the policy_val set of images.
- Of the remaining images, the images with a positive label for any attribute in the policy_train form the policy_train set of images.
- The remaining images form the set of policy_pretrain images.
This iterative procedure ensures that images in each of the policy_train, policy_val and policy_test result in the introduction of new attributes for which the classifier is not already trained. The data split will be included in the code release.

C Initial Retrieval

We wish to simulate dialogs as refinements of an initial retrieval based on the product description. At the start of each batch of interactions, for each description corresponding to an image in the current classifier_test subset, we rank all images in this subset according to a variant of the score in Equation 1. Instead of directly using classifier probabilities, we threshold the probabilities \( p_w(i) \) to obtain decisions \( d_w(i) \). The threshold for each attribute is chosen to maximize the F1 score for that attribute on the current set of validation images and labels. This is initially the classifier_test subset of the policy_pretrain set, and gets expanded with a fraction of the labels obtained using active learning queries. This F1 score is also used in the baseline static policy (section 4.8) and features for the learned dialog policies (sections 4.5, 4.6 and 4.7).

Also, while the initial belief in the dialogs (Equation 1) only assumes that attributes mentioned in the description are positive for the target image, in the retrieval phase, we additionally assume that attributes not mentioned in the description are negative. Then,

\[
b(i) = \prod_{w \in W_d} p_w(i) \prod_{w \not\in W_d} (1 - p_w(i)) \approx \prod_{w : d_w(i) = 1} c_1 \prod_{w : d_w(i) = 0} c_2
\]

\[
s(i) = \log b(i) = c_1 \{w : d_w(i) = 1\} + c_2 \{w : d_w(i) = 0\}
\]

We use the score \( s(i) \) to rank images, where \( c_1 \) and \( c_2 \) are hyperparameters tuned on the classifier_test subset of the policy_pretrain set. Our reported results use \( c_1 = 0.9 \) and \( c_2 = 0.1 \).

D Policy Representation and Learning

At any state \( s \), the agent can take one of the following actions:

- A special action \( a_g \) for guessing – the image is chosen using Equation 2.
- A set of clarification actions \( A_C \) – one for each attribute.
- A set of actions \( A_L \) corresponding to possible active learning queries – one example query per attribute and one label query corresponding to each pair \((w, i)\) for \( w \in W, i \in I^{Tr} \).

We learn a hierarchical dialog policy that consists of three parts as described below. For each policy, we obtain a feature representation of a state-action pair, as outlined in sections 4.5, 4.6 and 4.7.

- A clarification policy \( \pi_C \) to choose the best possible clarification action \( a_C^* \in A_C \) in the current state, using features \( \phi_C(s, a_C) \) for action \( a_C \) in state \( s \).
- An active learning policy \( \pi_{AL} \) to choose the best possible active learning query \( a_{AL}^* \in A_{AL} \) in the current state, using features \( \phi_{AL}(s, a_{AL}) \) for action \( a_{AL} \) in state \( s \). We reduce the action space of the active learning policy to one example query action per attribute, and one label query action per attribute corresponding to the image with probability closest to 0.5 for that attribute \((\argmin_{i \in I^{Tr}} |p_w(i) - 0.5|)\).
- A decision policy \( \pi_D \) that chooses between \( a_g, a_C^* \) and \( a_{AL}^* \), using features \( \phi_D(s, a) \) for action \( a \) in state \( s \).

We experiment with using both Q-learning and A3C [20] for policy learning. We use the same model structure for all three policies but no shared parameters. In the following discussion about the model structure for the policy, we will refer to state-action features \( \phi(s, a) \) and policy \( \pi(s, a) \), which is intended to represent the appropriate input and output for each policy.

For Q-learning we use a single-layer neural network with hidden layer size 100, whose input is the feature vector \( \pi(s, a) \), and output is the Q-value \( Q_{\pi}(s, a) \) of action \( a \) in state \( s \) under policy \( \pi \). Suppose action \( a \) is taken in state \( s \) resulting in reward \( r \) and next state \( s' \), we update the network with new targets:

\[
Q_{\pi}(s, a) = r + \gamma * \max_{a'} Q_{\pi}(s', a')
\]
In the policy training phase, we choose \( \epsilon \)-greedy actions with \( \epsilon = 0.1 \) and in the policy validation and testing phases, we choose actions greedily. We use \( \gamma = 1.0 \).

For A3C, as a critic, we use a network similar to Q-learning, predicting \( Q_\pi(s,a) \). The actor uses a policy representation:

\[
\pi_\theta(s,a) = \frac{e^{\theta^T \phi(s,a)}}{\sum_a e^{\theta^T \phi(s,a)}}
\]

where \( \theta \) is a learned parameter vector. Suppose action \( a \) is taken in state \( s \) resulting in reward \( r \) and next state \( s' \), the critic network is updated similar to Q-learning and the actor weights are updated as:

\[
\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(s,a) A_\pi(s,a)
\]

where \( A_\pi(s,a) = r + \gamma \sum_a \pi_\theta(s,a) Q_\pi(s,a) \)

where \( Q_\pi(s,a) \) is the estimate from the critic network. We use \( \gamma = 1.0 \) and \( \alpha = 0.01 \).

## E Estimation of Information Gain

Our static baseline for choosing clarification questions is based on prior work in goal-oriented dialog that attempts to estimate the information gain of a clarification question [17]. In this setting, the agent asking questions needs to identify a target object among a set of candidate objects, and can ask clarification questions to help identify the target. Let \( C, Q_t \) and \( A_t \) be random variables corresponding to the target object, question in turn \( t \) and answer in turn \( t \) respectively, and \( c, q_t \) and \( a_t \) represent specific values of these variables. Then the information gain from asking question \( q_t \), given previous questions \( q_{1:t-1} \) and their answers \( a_{1:t-1} \) is:

\[
I[C, A_t; q_t, a_{1:t-1}, q_{1:t-1}] = \sum_{a_t} \sum_c p(c|a_{1:t-1}, q_{1:t-1}) p(a_t|c, q_t, a_{1:t-1}, q_{1:t-1}) \ln \frac{p(a_t|c, q_t, a_{1:t-1}, q_{1:t-1})}{p(a_t|q_t, a_{1:t-1}, q_{1:t-1})}
\]

where \( p(a_t|q_t, a_{1:t-1}, q_{1:t-1}) = \sum_c p(c|a_{1:t-1}, q_{1:t-1}) p(a_t|c, q_t, a_{1:t-1}, q_{1:t-1}) \)

In our case, possible targets \( c \) correspond to possible images \( i \). As in prior work [17], \( p(c|a_{1:t-1}, q_{1:t-1}) \) corresponds to the estimated likelihood of target \( c \) given the conversation history, which in our case is \( b(i) \). We also make an additional assumption that the answer to question \( q_t \) depends only on the target image and not on prior questions and answers. Hence:

\[
p(a_t|c, q_t, a_{1:t-1}, q_{1:t-1}) = p(a_t|c, q_t)
\]

which in our case is \( P(a_t|q_t, i) \). Since our questions \( q_t \) are attributes and the attribute classifier is expected to provide the probability that attribute \( q \) is true for image \( i \), we get \( P(1|q_t, i) = p_q(i) \) and \( P(0|q_t, i) = 1 - p_q(i) \). In practice, we observe that the classifier does not produce well-calibrated probabilities despite the use of temperature correction, and we believe that this contributes to the poor performance of the static clarification policy.

Substituting these, we get information gain for question \( q \), which we represent using \( J(q) \), as:

\[
J(q) = \sum_{i \in T} \sum_{a \in \{0,1\}} b(i) P(a|q, i) \ln \left( \frac{P(a|q, i)}{\sum_i b(i) P(a|q, i)} \right)
\]

## F Classifier Design and Hyperparameters

For the attribute classifier, we initially experimented with alternate classifier designs such as binary SVMs using features extracted from Inception-V3 and fine-tuning Inception-V3 after altering the number of classes. We also experimented with alternate loss functions for fine-tuning Inception-V3, as well as the design in section [4.1] such as weighted cross entropy, and a ranking loss that maximizes the difference between the predicted probabilities of positive and negative attributes. Additionally, we
compared fine-tuning all layers of Inception-V3 with training/fine-tuning only the extra/final layers. We used Inception-V3 as the backbone network due to the results reported in the original paper [13]. However in contrast to the original paper, we found that our particular network design and loss function were required for obtaining reasonable classifier performance. Additionally, we found that it was required to initialize Inception-V3 with weights pretrained on ImageNet and train only the new layers on the iMaterialist dataset. These differences could be due to the differences in the data split. Our choice of data split results in many attributes always having a negative label during the training phase.

We also found that it was sometimes possible to obtain increases in the multilabel F1 metric proposed in the original paper [13] without any improvement on per attribute F1. For example, it is possible to obtain a multilabel F1 of 36.0 on the original validation set by identifying the 13 attributes with the largest number of positive examples, always predicting 1 for these, and always predicting 0 for the other attributes. Hence, we used the average per-attribute F1 to choose the design and hyperparameters of the classifier.

To initialize the classifier, we train for 100 epochs with a batch size of 8,192 and using RMSProp for optimization. We start with a learning rate of 0.1 which is decayed exponentially with a decay rate of 0.9 every 400 steps. For updating the classifier in between dialog batches, we use a batch size of 128 and perform a single epoch over images for which the label of at least one attribute has been updated.