Active Comparison Based Learning Incorporating User Uncertainty and Noise

Rachel Holladay
Robotics Institute
Carnegie Mellon University
rmh@andrew.cmu.edu

Shervin Javdani
Robotics Institute
Carnegie Mellon University
sjavdani@cmu.edu

Anca Dragan
EECS Department
University of California, Berkeley
anca@berkeley.edu

Siddhartha Srinivasa
Robotics Institute
Carnegie Mellon University
siddh@cs.cmu.edu

Abstract—**Our goal is to facilitate better human-robot collaboration by enabling robots to learn our preferences. To learn preferences, robots need to interact with users. We propose using comparison based learning, which learns preferences by asking a user to compare several alternatives. To minimize user burden, we use active learning. A challenge of comparison based learning is that it can be difficult for a user to say which item they prefer. Forcing the user to provide a preference in these cases leads to noisy responses, which increases the number of needed queries. Our key insight is that users can identify difficult comparisons and that we can use this information to learning their uncertainty. We present CLAUS (Comparison Learning Algorithm for Uncertain Situations), which model uncertainty and uses it to select and process comparison queries. Our user study suggests that CLAUS uses fewer queries than algorithms which force users to choose, while maintaining nearly the same accuracy.**

I. Introduction

We want to work alongside robots partners that understand and accommodate our preferences. To enable this, we need to properly communicate these preferences.

One method of communication is to directly interact with the robot, physically manipulating the robot to demonstrate the preferred way of executing a task [9, 7, 31]. However, it can be difficult and time consuming to provide demonstrations to complex robots [36, 17] and nearly impossible in domains such as swarm robotics [3]. To ease the former difficulty, previous methods include mixing trajectory demonstrations with keyframe poses [2] and using demonstrations to improve existing trajectories rather than provide optimal examples [21].

Moving away from demonstrations for preference learning, Daniel et. al [12] asked users to directly score robot performance. An alternate method is to ask users to rank paths or policies generated by the robot [17, 3].

Fürnkranz et. al [14] explored preference elicitation learning when it is difficult to provide a real-valued reward and instead easier to express a preference. This method of *comparison based learning* captures that it is easier for users to express these preferences relatively, rather than absolutely with a score.

Usually, comparison based learning presents the user with two examples and asks the user which item they prefer [10]. We assume the user's preferences are the result of optimizing some unknown cost function. These pairwise preferences induce a ranking in the space of possible cost functions [20]. This method of ranking or comparing for preference elicitation was found to be more efficient and as least as accurate, theoretically and empirically, as scoring individual labels [8].

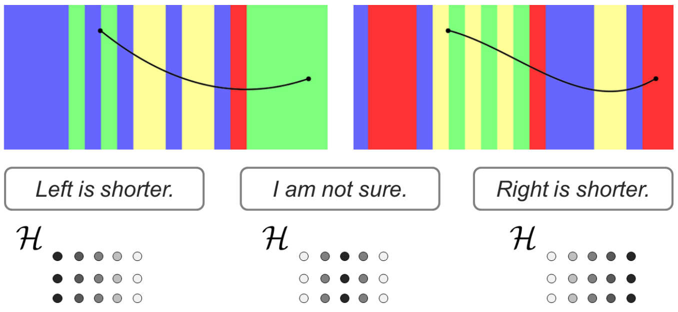


Fig. 1: An example query, asking a user to compare the lengths of two lines. Instead of forcing users to provide a comparison when they are not sure, we enable them to inform the learner about their uncertainty through the middle uncertainty button. The learner leverages this additional information in how it updates the probability distribution over hypotheses. This results in making fewer queries to the user.

Preference elicitation is often combined with active learning techniques. In active learning, the learner selects which training data to use as queries. This is in contrast to passive learning, where the learner receives whatever the user provides. In preference elicitation, active learning techniques can be used to reduce the number of queries the learner needs to make to the user [32]. Ailon [1] showed that active learning combined with pairwise preference elicitation produces almost optimal query complexity, performing strictly better than the passive alternative.

While active comparison based learning is efficient and user friendly, it needs to account for the fact that users are not perfect when making these comparison [25]. Nowak [29] handles noise by using a Bayesian model and assuming the user is only probably correct. Guillory and Blimes [18] deal with noise through repeat queries, eliminating a hypothesis only after a user has confirmed their preferences multiple times.

These approaches must make additional queries to account for noisy responses because they ask the user to make a decision when the user is uncertain. Not only are more queries burdensome to the user, but particular questions make it difficult to provide strict comparisons, frustrating the user [6]. Instead, users have reported that they would like to be able to express their uncertainty in difficult to judge situations [18].

Incorporating this, Guo and Sanner [19] presented a model where the user can express either that they prefer one item to another or are indifferent between the two items. When a user expresses indifference, this informs the model that the user's cost function of the two items is within some fixed value ϵ. Hence, if the cost of two items is within some small ϵ, the user is indifferent between the two. The model is evaluated with simulated users that follow the model's assumptions.

However, considering different people might have different ϵ's, it is nearly impossible to know the user's true ϵ in advance. We present a new method of learning in which we not only learn the user's cost function but also their level of uncertainty: their unique ϵ. Our model, CLAUS (Comparison Learning Algorithm for Uncertain Situations), accounts for the fact that users can identify when questions are difficult, i.e. when they are *uncertain*. Through a large user study, we show that modeling this user uncertainty allows the model to ask fewer queries.

As an example, perhaps the user prefers paths that are shorter for some task. Our goal is to learn that preference. If presented with the two paths in Fig.1, they may have a difficult time assessing which path they prefer. In CLAUS, we allow the user to specify that they are uncertain of their preference on these items. This indicates to our algorithm that user's cost function has similar costs for these items, providing useful feedback for learning.

Our work makes the following contributions:

1. CLAUS (Comparison Learning Algorithm for Uncertain Situations). We describe and evaluate a comparison based learning method that models user uncertainty, and utilizes a query-dependent noise model rooted in psychology research.

2. Formulation within an Active Learning Framework We frame CLAUS as an extension of EC^2, an active learning framework [16]. Importantly, this framework allows us to model uncertainty with a latent variable that need not be learned explicitly.

3. Evaluation with Real Users We conduct a large-scale user study that compared CLAUS to a method that did not model user uncertainty. Our results show that CLAUS required fewer queries to achieve nearly the same accuracy.

II. Problem Formulation

We assume that user preference corresponds to some cost function, $c^* \in \mathcal{C}$, where preferred items have lower cost. Our goal is to determine c^* with the fewest queries.

A. Setup

We cast our problem in the general framework of Bayesian active learning [16]. In this setting, we are given a discrete set of hypotheses $h \in \mathcal{H}$, and wish to distinguish among them by performing a sequence of tests $t \in \mathcal{T}$. Each test results in some observation $o \in \mathcal{O}$, giving us information about the hypotheses. The objective is to quickly determine the true hypothesis $h^* \in \mathcal{H}$.

In the simplest comparison based learning formulation, the hypotheses are the cost functions, $\mathcal{H} = \mathcal{C}$. Each test is composed of two items $t = (x, y)$ that the user compares. The user's response to a test corresponds to some observation $o \in \mathcal{O}$, e.g. indicating that $x >_{c^*} y$ or $x <_{c^*} y$. After performing m tests and receiving observations, our evidence is captured by the sequence of test-observation pairs $\mathcal{S} \subseteq \mathcal{T} \times \mathcal{O}$, where $\mathcal{S} = \{(t_1, o_1), \cdots, (t_m, o_m)\}$.

In the noiseless setting, this problem has been formulated as an adapative submodular maximization [15]. Here, an objective function f is defined that is maximized if and only if the true hypothesis has been determined. The goal is to maximize f with the fewest tests. Importantly, if f is adaptive submodular, then maximizing the expected gain in f with an efficient greedy algorithm produces a provably near-optimal solution [15]. This algorithm is given in Algorithm 1.

Algorithm 1 Greedy Test Selection

```
1: Given: Utility function $f$, Hypotheses $\mathcal{H}$, Tests $\mathcal{T}$
2: Initialize: Set Evidence so far $\mathcal{S} = \emptyset$
3: procedure WHILE NOTDONE
4:     $best\_test = \arg\max_{t \in \mathcal{T}} \mathbb{E}_o[f(\mathcal{S} \cup (t, o))]$
5:     Execute best_test, Receive Observation $o$
6:     $\mathcal{S} = \mathcal{S} \cup (t, o)$
```

If our goal is to determine $h^* \in \mathcal{H}$, and our test responses are noiseless, we can utilize the objective defined by Generalized Binary Search (GBS) [28]. We review this in the following section, and extend it to the noisy and uncertain settings.

B. Test Selection

In the noiseless setting, we can gather information by determining that hypotheses are *inconsistent* with the evidence, e.g. the cost function does not agree with the user's preference response. If a hypothesis is determined to be inconsistent, we say that it is removed. In GBS, the goal is to quickly find h^* by removing all other hypotheses.

Let $w(\mathcal{H} \mid \mathcal{S})$ correspond to the probability mass of all hypotheses still consistent with evidence. Maximizing the GBS objective f_{GBS} corresponds to maximizing the number of hypotheses removed:

$$f_{\text{GBS}}(\mathcal{S}) = w(\mathcal{H}) - w(\mathcal{H} \mid \mathcal{S})$$

The total weight of all the hypotheses, given our evidence, is the summed weight of all the hypotheses. The weight of each hypothesis is the probability of the evidence given that hypothesis.

$$w(\mathcal{H} \mid \mathcal{S}) = \sum_{h \in \mathcal{H}} w(h \mid \mathcal{S})$$

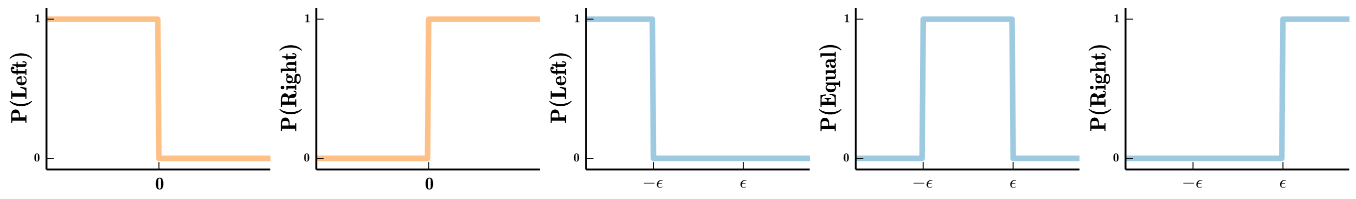
$$w(h \mid \mathcal{S}) = p(\mathcal{S} \mid h)$$

We assume that the sequence of test-observation pairs (t, o) in $\mathcal{S}$ is independent:

$$p(\mathcal{S} \mid h) = \prod_{(t,o) \in \mathcal{S}} p((t, o) \mid h)$$

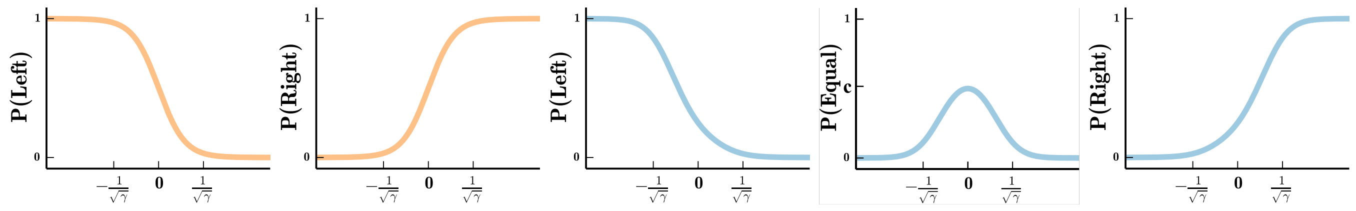
In the noiseless setting, we assume the user always selects the item that minimizes their cost function:

$$p((t, o = x) \mid h) = \begin{cases} 1 & c_h(x) < c_h(y) \\ 0 & \text{else} \end{cases}$$



(a) GBS (Left) (b) GBS (Right) (c) CLAUS (Left) (d) CLAUS (Equal) (e) CLAUS (Right)

Fig. 2: User response model in the noiseless setting



(a) GBS (Left) (b) GBS (Right) (c) CLAUS (Left) (d) CLAUS (Equal) (e) CLAUS (Right)

Fig. 3: Luce Sheppard noise model

C. Modeling User Noise

Unfortunately, users are not perfect evaluators of a cost function, and treating them as such leads to poor learning performance [6]. Many previous works account for noisy responses in active learning frameworks [18, 33, 23, 29]. The majority of these models apply the same level of noise to all queries, e.g. constant probability of a noisy response.

While these models can lead to impressive theoretical results [16, 18, 11], they do not accurately reflect real-world human behavior [13, 34]. Instead, Du and Ling [13] suggest that noise be *query-dependent*; that for some queries the user will be confident and relatively noiseless, while in other situations the user could be very unsure and therefore very noisy. For comparison based learning, this suggests that noise be related to the difference in costs of items in the query.

This query-dependent noise is supported in the psychology literature. In particular, the Luce-Sheppard Choice Rule states that users are evaluating a cost function subject to this kind of noise [34, 27, 26]. Viappiani and Boutilier [35] derive a logistic model based on the Luce-Sheppard Choice Rule, thus creating a noise model for this paradigm. Akrour et al. [5] formulate a similar query-dependent noise model, though they don't utilize the logistic model.

We model noise based on the Luce-Sheppard Choice Rule where the probability of observation for a given test is:

$$p((t, o = x) \mid h) \propto \exp\left(-\gamma c_h(x)\right)$$

D. CLAUS

Most comparison based learning methods require that users perform strict comparisons, stating that one item is better than another [20, 8, 10, 30, 18, 24, 4, 5]. But what should the user do when the items being compared are nearly equivalent? In the case of strict comparisons, the user is likely to provide a noisy response. While there have been noise models developed to alleviate this, it often leads to additional queries [4, 5].

In prior works, users have reported that they often have difficulty making a comparison, and would like the ability to indicate uncertainty [18]. We hypothesize that allowing the user to express their uncertainty will both increase user satisfaction, and increase the efficiency of our learning algorithm.

We present *CLAUS*, Comparison Learning Algorithm for Uncertain Situations, which models and accounts for that fact that users will be uncertain about their preference when the cost of the items being compared is close.

We model user uncertainty through a new parameter $\epsilon \in E$. Just as each user has their own cost function c^*, we assume they have their own ϵ^*, such that if $|c(x) - c(y)| < \epsilon$ the user is uncertain. Importantly, if the user expresses uncertainty, this indicates that the user's cost function c^* assigns similar cost to the items, enabling us to gather information about cost functions.

In order to allow the user to express this uncertainty, we have to modify our queries. Our tests still correspond to showing the user two items, $t = (x, y)$. However, we now have three possible observations: $\mathcal{O}(t) = (x, y, \widetilde{xy})$, where $\widetilde{xy}$ corresponds to expressing uncertainty.

To utilize this uncertainty model, we modify our hypotheses to include ϵ, such that $\mathcal{H} \subseteq \mathcal{C} \times E$. Naively applying GBS in this setting would correspond to learning a (c^*, ϵ^*) pair. While ϵ is necessary for modelling the user responses, our objective only corresponds to learning c^*. Learning ϵ^* exactly may cause us to perform more queries then necessary. Instead we want to find the user's c^* for any viable ϵ.

We cast this problem as one of Equivalence Class Determination (ECD) [16], where the goal is not to determine the true hypothesis, but rather to determine some equivalence class. Each equivalence class corresponds to a set of hypotheses that we do not need to distinguish between. We say an equivalence class is determined when all remaining hypotheses are within a single equivalence class. We adapt this framework in our setting by creating an equivalence class for each cost function c, to which we assign all hypotheses that have that cost function c, and any ϵ.

To solve ECD, Golovin et al. introduced EC^2 [16]. This algorithm defines weighted edges between hypotheses in

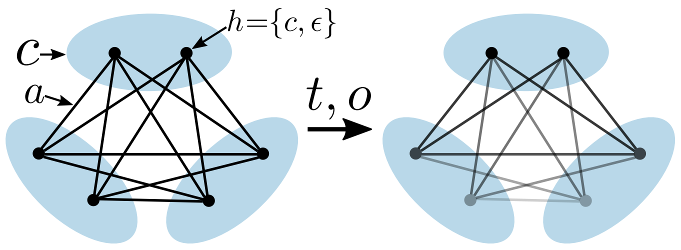


Fig. 4: CLAUS using EC^2. Each cost function c corresponds to an equivalence class (blue ellipse). Hypotheses (black dots) are $\{c, \epsilon\}$ pairs. Hypotheses sharing a cost c are said to be inside the equivalence class of c. The algorithm constructs a graph, drawing an edge (black line) between hypotheses in different equivalence classes. After performing a test and receiving an observation, the evidence results in downweighting some hypotheses, which in turn downweights the edges they connect to.

different equivalence classes. We say that an edge is cut when either hypothesis it is connected to is removed. By construction, we determine an equivalence class if and only all edges have been cut. Fig.4 provides a visualization of applying EC^{2}to CLAUS .

We define a new objective function, f_{CLAUS}, that optimizes over edges, $a \in \mathcal{A}$

$$f_{\text{CLAUS}}(\mathcal{S}) = w(\mathcal{A}) - w(\mathcal{A} \mid \mathcal{S})$$

The weight of all the edges given the evidence, is the sum of weights over the edges:

$$w(\mathcal{A} \mid \mathcal{S}) = \sum_{a \in \mathcal{A}} w(a \mid \mathcal{S})$$

The weight of an edge is the product of the weights of the hypotheses connected by that edge:

$$w(a \mid \mathcal{S}) = \prod_{h \in a} w(h \mid \mathcal{S})$$

From here $w(h \mid \mathcal{S})$ is defined as it was in Sec. II-B.

Let c_h be the cost function of the hypothesis, and ϵ_h is the uncertainty parameter. In the noiseless setting, we assume user response corresponds to:

$$p((t, o = x) \mid h) = \begin{cases} 1 & c_h(x) < c_h(y) - \epsilon_h \\ 0 & \text{else} \end{cases}$$
$$p((t, o = \widetilde{xy}) \mid h) = \begin{cases} 1 & |c_h(x) - c_h(y)|^2 < \epsilon_h^2 \\ 0 & \text{else} \end{cases}$$

We extend this to model noise, stemming from the Luce-Sheppard model from Sec. II-C.

$$p((t, o = x) \mid h) \propto \exp\left(-\gamma(c_h(x) - c_h(y))\right)$$
$$p((t, o = \widetilde{xy}) \mid h) \propto \exp\left(-\frac{1}{\epsilon_h^2}\left[c_h(x) - c_h(y)\right]^2\right) c$$

Naively, we could compute this objective by iterating over every pair of hypotheses. Thankfully, we do not need to do so, as this computation is equivalent to computing an elementary symmetric polynomial of order 2, which can be computed in time linear in the number of hypotheses [16, 22, 11].

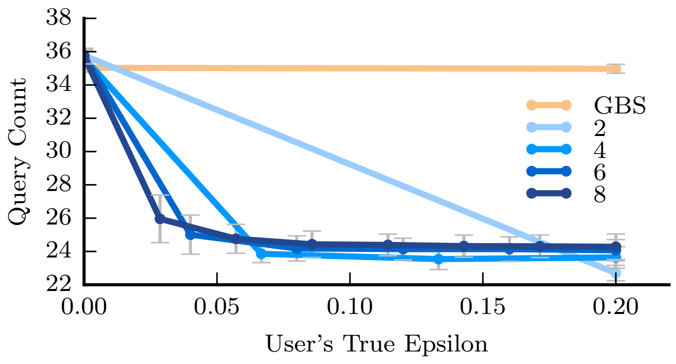


Fig. 5: We compare the affect of the number of epsilons on CLAUS's query count across the user's ϵ^*. We also show the query count of GBS. Note that when the user expresses any uncertainty, $\epsilon^* > 0$, all CLAUS methods utilize far fewer queries than GBS.

III. GBS vs CLAUS : Simulation Experiments

We conduct several simulation experiments to test how altering our noise and uncertainty parameters affects the quality of the learning algorithms. Specifically we examined three parameters: the number of epsilons, the noise parameter for GBS and one of the noise parameters for CLAUS.

In CLAUS, we capture uncertainty through an epsilon parameter, such that if two items are ϵ-close in cost, we model the user as uncertain. For our algorithm, we first generate the discrete set of epsilons E by linearly interpolating between a minimum and maximum value. To test the effect of E on our learning algorithm, we varied the interpolation rate and recorded the query count for noiseless CLAUS across 100 trials each (Fig.5). As the number of epsilons increases, the learning algorithm requires only slightly more queries, thanks to the use of EC^{2}which does not require learning the exact ϵ.

In noisy GBS and CLAUS, γ models the noise in the user's responses when selecting one test item. In order to evaluate how robust the algorithm is to this parameter, we simulate user responses with γ^{u}, and seed the model in our noisy GBS algorithm with a different γ^{l}. We compare the accuracy and query count in Fig.6. We see that the higher γ^{l} generally corresponds to higher accuracy, though it also requires a slightly higher query count. Nonetheless, the results suggest it is better to model too much noise then not enough.

In noisy CLAUS, c models how likely the user is to express uncertainty. In order to evaluate how robust the algorithm is to this parameter, we simulate user responses with c^{u}, and seed the model in our noisy CLAUS algorithm with a different c^{l}. We compare the accuracy and query count in Fig.6. We see that generally, it's preferable to underestimate the c parameter, which corresponds to underestimating how often the user will express uncertainty. However, there is a lower limit as if $\gamma = 0$ then the model becomes GBS, since the model expects the user to never express uncertainty.

For the user study in Sec. IV, we ran a small pilot study and fit the noise parameters to user responses.

IV. User Evaluation of GBS and CLAUS

We compare GBS and CLAUS using our query-dependent noise model (Sec. II-C) with real users. One challenge in eval-

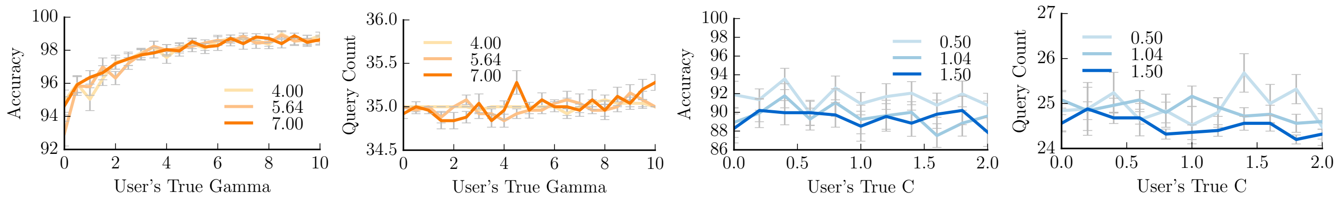


Fig. 6: As part of our simulation experiments, we evaluate the robustness of our noise parameters. In the left two figures, we see the effect of varying the learner's γ as compared to the user's true γ. On the right is the effect of varying the learner's c compared to the user's true c.

uating user preference learning is selecting a good evaluation metric. User responses are noisy, making it difficult to get ground truth test data to evaluate the quality of the learned model.

To circumvent this and focus our assessment on the effect of modeling noisy user responses and uncertainty, we ask users to evaluate a well defined objective for which we know the correct answer. For each query, users were shown two images of black lines on a colorful background, and asked which line is shorter. The set of possible cost functions $\mathcal{C}$ modeled both the line length and the background colors. Our objective was to learn the cost function $c^* \in \mathcal{C}$ that assigned cost only to line length, and ignored colors.

A. Experimental Setup

Procedure.

For both GBS and CLAUS, we randomly sample $|\mathcal{C}| = 10000$ linear cost functions of the form $c(x) = w^T \phi_x$, where ϕ_x corresponds to features dependent on path length and color. For EC^2, we also sample $|E| = 6$ epsilons, such that $|\mathcal{H}| = |E| \times |\mathcal{C}|$.

Users used GBS and CLAUS in a randomized order. When using GBS, users could indicate which picture they believed had the shorter line with buttons 'Left' and 'Right'. When using CLAUS, users had an additional button to express uncertainty. We will discuss below how we test different phrasings for this uncertainty button.

The study concluded with a few survey question comparing experiences with each algorithm.

In the noiseless setting, we terminate the algorithm when all hypotheses consistent with the evidence correspond to only one $c \in \mathcal{C}$. In the noisy setting, hypotheses are never declared inconsistent, but only change in distribution. Therefore, we select some threshold for our objective function given evidence $f(\mathcal{S})$ to terminate on.

Additionally, the classic GBS objective maximizes the number of hypotheses declared inconsistent. However, we are more interested in distinguishing between different cost functions, an objective better captured by EC^2. While these are achieved at the same time in the noiseless setting, they are different when modeling noise. In our experiments, instead of using classic GBS, we utilize EC^{2}where each cost function is assigned to its own region, as this better captures our objective.

Hypotheses.

H1. The wording choice of the CLAUS uncertainty button effects user's preferences and performance.

H2. CLAUS, with the better performing uncertainty button labeling, will require fewer queries than GBS.

H3. CLAUS, with the better performing uncertainty button labeling, will be as accurate as GBS.

Manipulated Factor. Our study had two manipulated factors: algorithm and uncertainity labeling. The algorithms used were GBS and CLAUS, implemented as described above.

With respect to the label of the uncertainty response, our pilot studies revealed that different labels can lead to different interpretations of what level of uncertainty is appropriate for such a response. **H1** states that how this uncertainty is labeled and presented to the user will effect how the user views and uses it. This, in turn, can effect performance. We decided to test two alternatives: 'About Equal' and 'I am not Sure'.

These capture two perspectives about uncertainty. 'About Equal' communicates that the two items compared are so close that the user cannot distinguish them. Hence, in the binary setting, the user would be uncertain of which is better. 'I am not sure' communicates that the user does not know how to judge the two items or feels uncertain doing so. In the binary case, the user would have otherwise felt at ease and probably provided a noisy answer.

In both cases, for CLAUS with 'About Equal' and CLAUS with 'I am not Sure', the algorithm remained unchanged.

Participants. We used a mixed design for subject allocation. The algorithm factor was within-subjects, meaning that every user saw both GBS and CLAUS. We chose this design so that users can directly compare between the two. The uncertainty labeling factor was between-subjects. We did this to insure that the users' understanding of uncertainty would not be biased by what they saw first.

We recruited a total of 120 users (30 per condition) on Amazon's Mechanical Turk. After eliminating users who answered a control question incorrectly, we had 111 users (56 female, 55 male, aged 18-69).

B. Analysis

The results regarding query count and accuracy, split by the dependent measures, are shown in Table I and the CLAUS parameters, such as the ϵ value and the number of times user's expressed uncertainty, are shown in Table II.

With respect to **H1** we wish to see if the labeling of the CLAUS uncertainty button effects performance, comparing 'About Equal' to 'I am not Sure'. If one performs better than the other we will declare this CLAUS labeling the best. Then we will compare this best CLAUS labeling with GBS in terms of the same performance metrics. Our performance metrics are: user preference, as captured in our Likert data, query count and accuracy.

In eliciting user preference we asked four Likert questions where 1 corresponded to a preference of GBS and 7 corresponded to a preference for CLAUS, with which ever label the participant used. We asked which method the user enjoyed, preferred, felt would learn best and trusted. The results can be seen in Fig.7.

User responses showed that many people preferred the variants of CLAUS to GBS since expressing their uncertainty made them feel more confident in their answer, and gave "some leeway, rather than a binary decision". Some users reported feeling "frustrated" when they couldn't state the lines where similar in length and "didn't like being restricted".

Those who liked GBS better enjoyed fewer buttons and liked that they were forced into make a decision, although they did admit it was harder and coerced them into examining the query "with a sharper eye".

With respect to **H1**, a t-test showed a marginal decrease in query count for the "About Equal" option, ($t(110) = 1.77, p = .0794$). For accuracy, an equivalence test using TOST and a 3% threshold showed that the two options produce equivalent accuracies, $p = .0179$.

The subjective measures were correlated, with Cronbach's $\alpha = .84$. The effect of the question text on the combined measure was not significant ($t(110) = .65, p = .5144$). An equivalence test using TOST showed that the two options produce preferences within 1 Likert point of each other, $p = .0152$.

While the two manipulated factors were approximately equal with respect to accuracy and preference, since 'About Equal' had a smaller query count, we determined this to be the better labeling and proceeded to compare its performance to GBS .

For **H2**, a repeated measures ANOVA for query count resulted in a good fit of the data (adjusted $R^2 = .85$) and showed that CLAUS significantly reduced the number of queries compared to GBS ($F(1,117) = 632.11, p < .0001$). The reduction compared to responses is plotted in Fig.8.

For **H3**, a repeated measures ANOVA for accuracy did not yield as good of a fit (adjusted $R^2 = .19$). It did suggest that CLAUS had lower accuracy than GBS ($F(1,117) = 13.64, p < .001$), reducing accuracy from a least squares mean of 94.74 for GBS to 91.11 for CLAUS. Despite the poor fit which indicates we have to take this result with a grain of salt, it seems that reducing query count can come at the expense of slight loss of accuracy.

TABLE I: Accuracy and Query Count

Category	Accuracy	Query Count
GBS - About Equal	94.15 ± 0.52	36.02 ± 0.03
GBS - Not Sure	$\mathbf{94.66 \pm 0.55}$	35.95 ± 0.04
CLAUS - About Equal	91.56 ± 0.84	$\mathbf{25.93 \pm 0.41}$
CLAUS - Not Sure	90.86 ± 0.74	26.98 ± 0.47

TABLE II: CLAUS Parameters

Category	Marked Uncertainty	Epsilon
About Equal	7.80 ± 0.70	0.07 ± 0.00
Not Sure	5.57 ± 0.71	0.06 ± 0.01

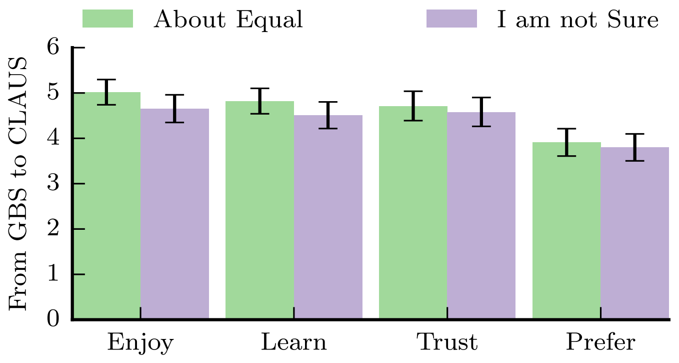


Fig. 7: The Likert data from the study showed that users narrowly favored 'About Equal' to 'I am not sure'

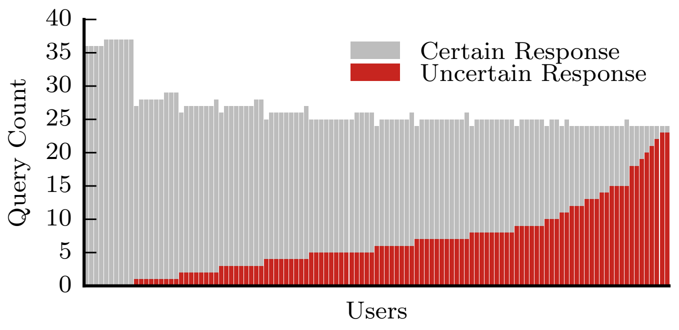


Fig. 8: We categorize 'Certain Responses' as 'Left' or 'Right' while an 'Uncertain Responses' as either uncertainty labeling. CLAUS uses the additional information provided by each uncertain response to make fewer queries.

V. Conclusion

Comparison based learning traditionally forces users to make comparisons between two options, even when those two options are hard to compare and users would be uncertain in their answer. Our key insight is that users can identify that they are uncertain, and that we can leverage this additional information to design learners that make fewer queries. We introduced *CLAUS*, which models user uncertainty as occurring when the two options are within an ϵ difference in cost. It then uses this model when deciding on queries and on reweighing/removing hypothesized cost functions.

We also account for user noise, drawing on previous work and psychology models to develop our noise model. We ran a user study, with results suggesting that our method asks fewer queries while achieving almost the same accuracy level, as compared to GBS.

We acknowledge that our uncertainty and noise models may not perfectly model humans. However, their performance suggests that despite their simplicity, accounting for human behavior can lead to better human-in-the-loop machine learning techniques.

Acknowledgments

This material is based upon work supported by NSF 1227495 and CMU's SRC URO's undergraduate research grants. We thank the members of the Personal Robotics Lab for helpful discussion and advice.

References

[1] Nir Ailon. Active learning ranking from pairwise preferences with almost optimal query complexity. In *NIPS*, pages 810–818, 2011.

[2] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L Thomaz. Keyframe-based learning from demonstration. *IJSR*, 4(4):343–355, 2012.

[3] Riad Akrour, Marc Schoenauer, and Michele Sebag. Preference-based policy learning. In *ecmlpkdd*, pages 12–27. Springer, 2011.

[4] Riad Akrour, Marc Schoenauer, and Michèle Sebag. April: Active preference learning-based reinforcement learning. In *ECMLPKDD*, 2012.

[5] Riad Akrour, Marc Schoenauer, , Jean-Christophe Souplet, and Michèle Sebag. Programming by feedback. In *ICML*, 2014.

[6] Saleema Amershi, Maya Cakmak, W. Bradley Knox, and Todd Kulesza. Power to the people: The role of humans in interactive machine learning. *AI Magazine*, 2014.

[7] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57 (5):469–483, 2009.

[8] Klaus Brinker, Johannes Fürnkranz, and Eyke Hüllermeier. Label ranking by learning pairwise preferences. Technical report, Citeseer, 2007.

[9] Sylvain Calinon and Aude Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 255–262. ACM, 2007.

[10] Ben Carterette, Paul N. Bennett, David Maxwell Chickering, and Susan T. Dumais. Here or there: Preference judgments for relevance. In *ECIR*, 2008.

[11] Yuxin Chen, Shervin Javdani, Amin Karbasi, J Andrew Bagnell, Siddhartha S Srinivasa, and Andreas Krause. Submodular surrogates for value of information. In *AAAI*, pages 3511–3518, 2015.

[12] Christian Daniel, Malte Viering, Jan Metz, Oliver Kroemer, and Jan Peters. Active reward learning. In *RSS*, 2014.

[13] Jun Du and Charles X Ling. Active learning with human-like noisy oracle. In *ICDM*, pages 797–802. IEEE, 2010.

[14] Johannes Fürnkranz, Eyke Hüllermeier, Weiwei Cheng, and Sang-Hyeun Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine learning*, 89(1-2):123–156, 2012.

[15] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *JAIR*, pages 427–486, 2011.

[16] Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In *NIPS*, pages 766–774, 2010.

[17] Artem Gritsenko and Dmitry Berenson. Learning task-specific path-quality cost functions from expert preferences. 2014.

[18] Andrew Guillory and Jeff Bilmes. Simultaneous learning and covering with adversarial noise. In *ICML*, 2011.

[19] Shengbo Guo and Scott Sanner. Real-time multiattribute bayesian preference elicitation with pairwise comparison queries. In *AISTATS*, pages 289–296, 2010.

[20] Eyke Hüllermeier and Johannes Fürnkranz. Comparison of ranking procedures in pairwise preference learning. In *IPMU*, 2004.

[21] Ashesh Jain, Brian Wojcik, Thorsten Joachims, and Ashutosh Saxena. Learning trajectory preferences for manipulators via iterative improvement. In *NIPS*, 2013.

[22] Shervin Javdani, Yuxin Chen, Amin Karbasi, Andreas Krause, J Andrew Bagnell, and Siddhartha Srinivasa. Near optimal bayesian active learning for decision making. In *AISTATS*, 2014.

[23] Matti Kääriäinen. Active learning in the non-realizable case. In *Algorithmic Learning Theory*, pages 63–77. Springer, 2006.

[24] Amin Karbasi, Stratis Ioannidis, and Laurent Massoulie. Comparison-based learning with rank nets. In *ICML*, 2012.

[25] Todd Kulesza, Saleema Amershi, Rich Caruana, Danyel Fisher, and Denis Charles. Structured labeling for facilitating concept evolution in machine learning. In *CHI*, 2014.

[26] Christopher G Lucas, Thomas L Griffiths, Fei Xu, and Christine Fawcett. A rational model of preference learning and choice prediction by children. In *NIPS*, pages 985–992, 2009.

[27] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2005.

[28] Robert Nowak. Generalized binary search. In *Communication, Control, and Computing, Allerton Conference on*, pages 568–574. IEEE, 2008.

[29] Robert Nowak. Noisy generalized binary search. In *NIPS*, pages 1366–1374, 2009.

[30] Kira Radinsky and Nir Ailon. Ranking from pairs and triplets: information quality, evaluation methods and query complexity. In *WSDM*, pages 105–114. ACM, 2011.

[31] Stefan Schaal. Learning from demonstration. *Advances in neural information processing systems*, pages 1040–1046, 1997.

[32] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.

[33] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *SIGKDD*, pages 614–622. ACM, 2008.

[34] Roger N Shepard. Stimulus and response generalization: A stochastic model relating generalization to distance in psychological space. *Psychometrika*, 22(4):325–345, 1957.

[35] Paolo Viappiani and Craig Boutilier. Optimal bayesian recommendation sets and myopically optimal choice query sets. In *NIPS*, pages 2352–2360, 2010.

[36] Aaron Wilson, Alan Fern, and Prasad Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In *NIPS*, 2012.